83 20.A.10

1720A
Instrument Controller

Volume II

FLUKE®

1720A
Instrument Controller

Manual Set Contents

VOLUME I

1720A User Manual
Set RS-232 Utility (SET)
File Utility (FUP)
Time and Date Utility (TIME)
Console Monitor (CONMON)
Floppy Disk Operating System (FDOS)
1720A Fluke BASIC Programming Manual
1720A Fluke Enhanced BASIC Reference Guide

VOLUME II

1720A Operator Manual
1720A Service Manual
1720A Controller Programming Worksheet
1720A Training Package
Disk Holder

Form No. E671 1/82

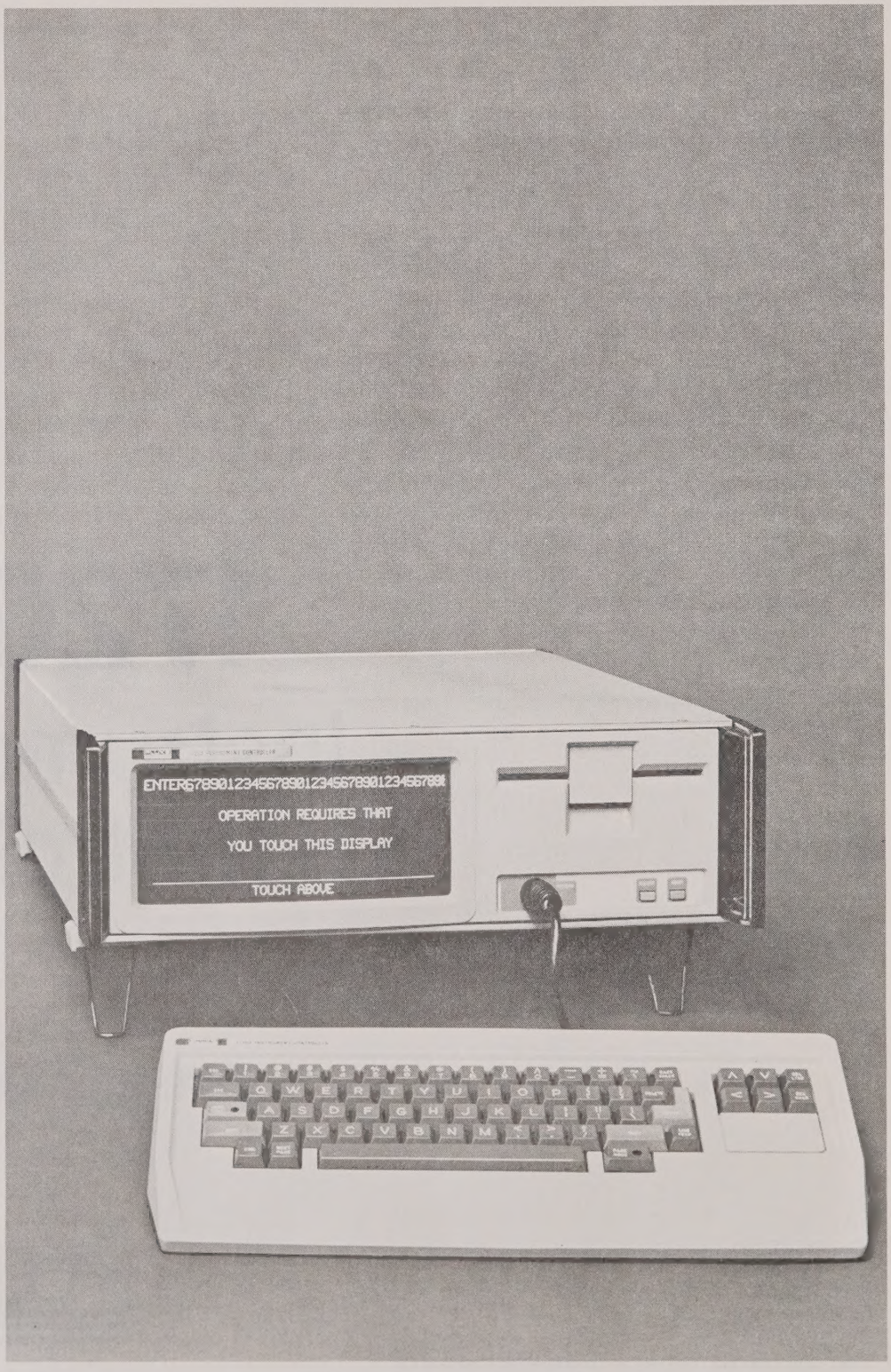
1720A
Instrument Controller

Operator Manual

P/N 518647
February 1980
Rev. 1 7/80

©1981, John Fluke Mfg. Co., Inc., all rights reserved Litho in U.S.A.

FLUKE®



1720A Instrument Controller

Digitized by the Internet Archive
in 2025

https://archive.org/details/manualsplus_08248

1720A
Operator Manual

INTRODUCTION

The 1720A instrument controller is designed for you, the operator. We have designed it so you can control instruments easily and clearly without having to be a computer programmer or electronics engineer. It has a minimum of controls and adjustments, not even a keyboard. Starting a procedure is a simple matter of inserting a flexible disk and turning on the power. The controller will even tell you when it's not working correctly. The touch sensitive display allows the 1720A to be programmed so your choices while conducting a test procedure will be simple, quick, and accurate.

This book will be your guide to using the controller. It contains information on start-up procedures, safety precautions, practical information on the modules that make up the controller and information on possible system errors.

GENERAL SAFETY REQUIREMENTS

WARNING

TO AVOID ELECTRICAL SHOCK HAZARDS, THE 1720A CONTROLLER MUST BE ADEQUATELY GROUNDED, EITHER WITH THE STANDARD GROUND PLUG OR THROUGH THE SYSTEM GROUND ON THE REAR OF THE CONTROLLER. ALWAYS CONNECT THE POWER CORD TO A GROUNDED OUTLET.

OPERATING FEATURES

The controls, indicators, and connectors for the 1720A are shown in Figure 1 and described in Table 1.

OPERATING NOTES

The following paragraphs describe various operator functions that you should consider before operating the 1720A.

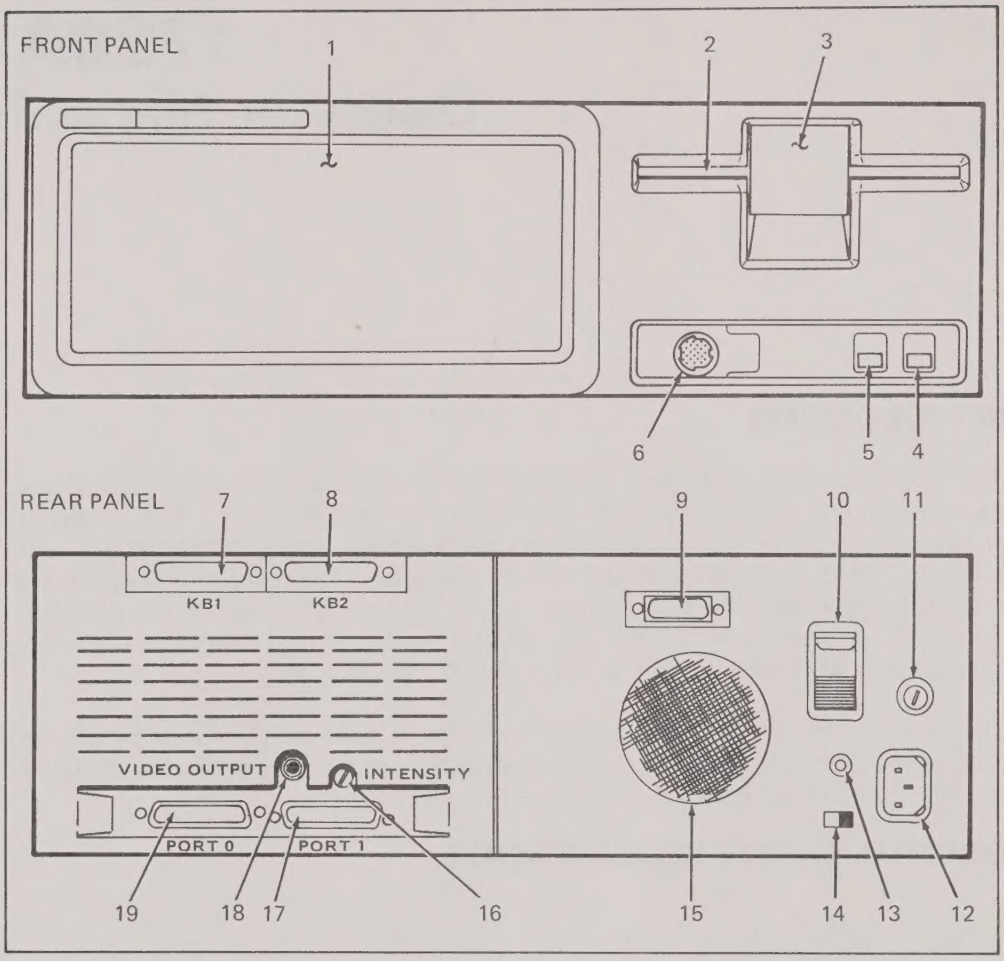


Figure 1. Front and Rear Panel Controls, Indicators and Connectors

Table 1. Front and Rear Panel Controls, Indicators and Connectors

REF. NO.	NAME	FUNCTION
1	TOUCH SENSITIVE DISPLAY	A display with touch sensitive areas which allows operation of a program by touch according to commands shown on the screen. The commands are program controlled and generated.
2	Disk Loading Slot	The floppy disk is loaded into the disk loading slot for program execution and data storage. See "Loading the Disk" for details.
3	Disk Entry Latch	Seats disk for operation.
4	ABORT pushbutton switch	Clears devices on the IEEE-488 buses and either stops the program or allows automatic program continuation, depending on your application program.
5	RESTART pushbutton switch	Initiates self test, clears all of main memory, and re-loads program.

Table 1. Front and Rear Panel Controls, Indicators and Connectors (cont)

REF. NO.	NAME	FUNCTION
6	Keyboard Connector	For connecting the detachable keyboard.
7	RS-232-C Connector #1	Standard I/O port for peripheral devices.
8	RS-232-C Connector #2	Standard I/O port for peripheral devices.
9	Remote Interface Connector	Provides for external E-Disk battery back-up, external control of ABORT and RESTART switches, and two software controlled lines.
10	Main Power Switch	Applies and removes line power to the controller.
11	Fuse	Power supply fuse.
12	Power Connector	Input line power connector.
13	System Ground	A common ground for system components.
14	E-Disk Enable Switch	Provides battery back-up for E-Disk
15	Air Filter	Filters air used to cool power supply. See maintenance procedures.
16	Display Intensity Control	A screwdriver adjust which controls the brightness of the display.
17	IEEE-488 Port 1 Connector	Standard IEEE-488-1978 Connector for instrumentation.
18	Composite Video Connector	Drives standard data display monitor.
19	IEEE-488 Port 0 Connector	Standard IEEE-488-1978 Connector for instrumentation.

Ventilation

The controller must have adequate ventilation to maintain its operating temperature. Keep it at least 4 inches, (12 cm), from a wall so that the fan may cool it properly. There is no minimum distance from top or bottom placement as the controller is designed to be mounted in a rack with other instruments.

Display Intensity

Adjust the display intensity for ease of viewing. The phosphor coating in the display will last longer at lower intensities.

Environmental Considerations

The operating range of the controller is 10°C to 40°C (50°F to 104°F), 8% to 80% humidity. Operation below 8% humidity can cause static electric discharges in the disk which may damage it and other components in the controller.

Electronic-Disk and Floppy Disk

If you turn off the controller while a test or procedure is in progress, you may lose the information being written on the floppy disk or on the Electronic-Disk, if included. Leave the system running while it is in a write operation.

E-Disk Enable Switch

The E-Disk enable switch must be in the enable mode for battery back-up of the E-Disk, should external power be interrupted.

NOTE

Battery back-up for the E-Disk will come on if the E-Disk enable switch on the controller is not in the disable mode when the system power is turned off.

System Shutdown

The controller will automatically shutdown under any one of the following conditions:

1. Internal temperature exceeds 70°C (158°F).
2. Overcurrent (controller is drawing too much current and may be overheating).
3. Overvoltage (controller regulated voltage is above limits).
4. Low Line (line voltage is below 90 volts or 180 volts).

The Flexible Disks

LOADING THE DISK

1. Open the disk slot latch by pressing in on the top of the square panel on the disk loading slot. (See Figure 2.)

2. Insert the disk as shown in Figure 3, label side up.

3. Gently relatch the disk slot by partially closing it several times to insure seating, then close it fully.

CAUTION

If the disk-entry latch is closed when the disk is improperly seated, the disk center hole may be damaged. This can be prevented by partially lowering and raising the latch to seat the drive hub prior to total closure of the disk-entry latch.

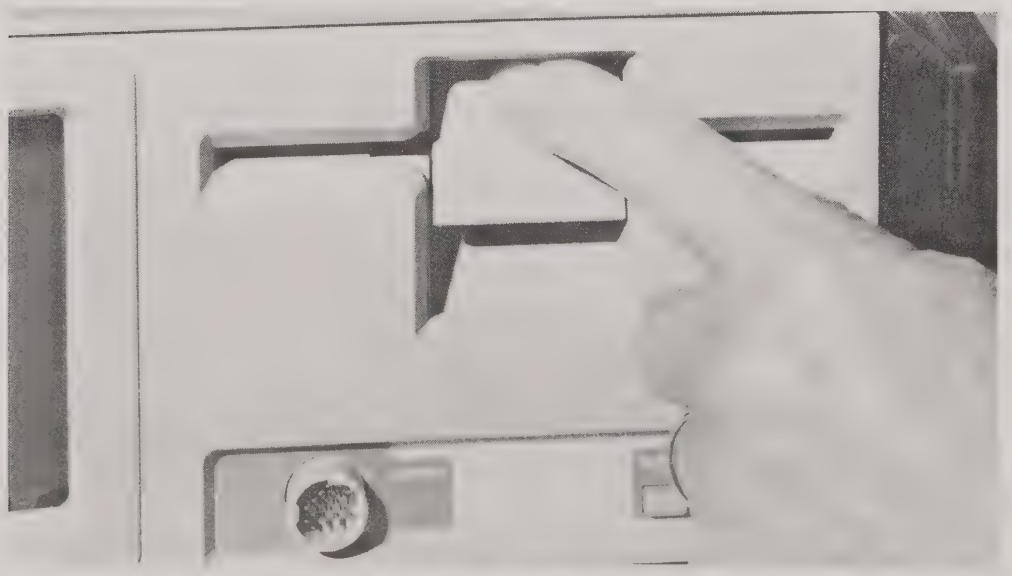


Figure 2. Opening the Disk Slot Latch

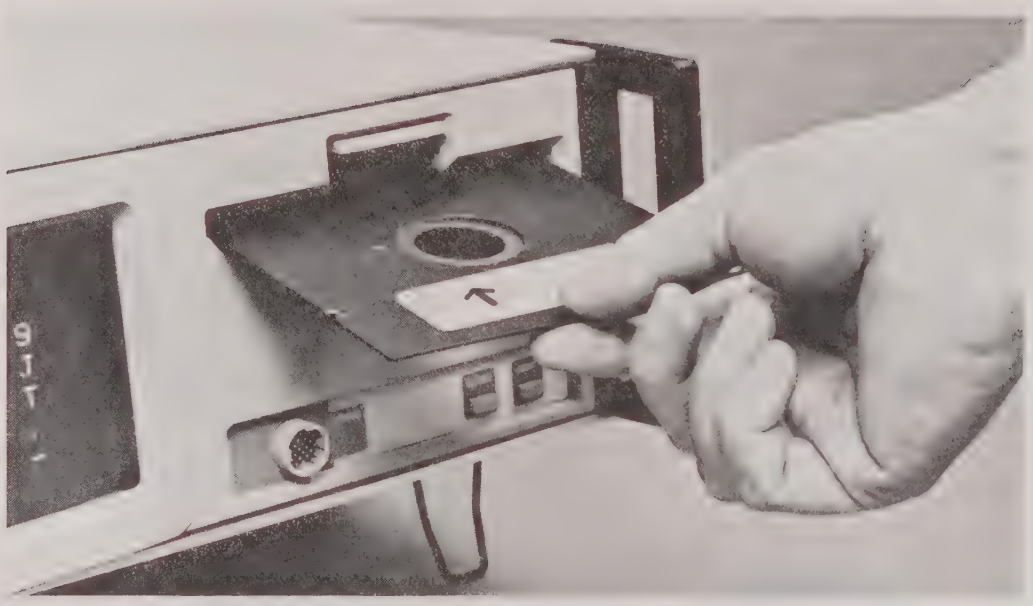


Figure 3. Inserting the Disk

CARE AND HANDLING (See Figure 4)

1. Always keep the disk in its protective folder when not in use.

2. The disk surface can be easily damaged. Do not touch the exposed disk surface with your hands or any material which may damage it.

3. Careless handling of the disks may damage them. Avoid dropping, throwing, or twisting the disks.

4. Store the disks vertically and support them so they will not bend. Stacking may distort them and affect their contents by pressing the side covers into the disk. Also avoid placing heavy objects on the disks.

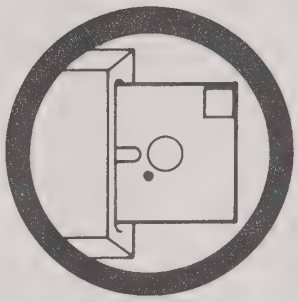
5. Direct sunlight may warp the disks. Protect them from the heat of the sun.

6. Magnetic sources may erase data on the disks. Keep them away from electric motors, generators, and transformers.

7. Mark disk labels before placing them on the disk envelope or use a felt tip pen. Pressure from a ball point pen or pencil through the envelope may destroy the data on the disk.

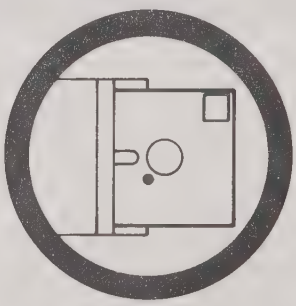
8. Temperature affects the disk's performance. Do not use a disk that is overheated or overcooled until it has reached room temperature.

DO NOT TOUCH
PRECISION SURFACE
WITH YOUR FINGERS.



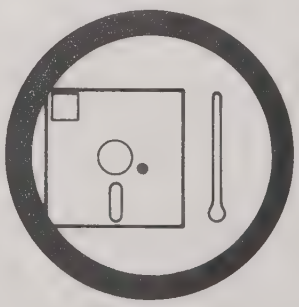
TO AVOID DAMAGE TO THE
DISKETTE AND TO YOUR DRIVE,
INSERT DISKETTE CAREFULLY.

KEEP DISKETTE AWAY
FROM MAGNETIC
FIELDS.



RETURN THE DISKETTE
TO ITS JACKET WHEN
NOT IN USE.

HANDLE WITH CARE
BENDING AND
FOLDING MAY
DAMAGE DISKETTE.



DISKETTES SHOULD
BE STORED AT
10° TO 52° C.
50° TO 125° F.

Figure 4. Care and Handling of the Disk

USE OF CONTROLS

The controls you will use are:

Touch Sensitive Display: touch sensitive display panel can be programmed to operate by simple commands, individual letters, numbers, or areas which must be touched one at a time. The programmer will write system programs to display information on the screen that may require your response. Touch the screen where indicated to respond to system queries. (See Figure 5.) If you view the display at too great an angle, the resulting parallax may cause you to touch the wrong area. Make sure you are facing the screen directly when making choices. Figure 6 illustrates the importance of the viewing angle when operating the touch sensitive display keyboard.

RESTART: Initiates SELF TEST, clears all of main memory and reloads programs.

ABORT: Clears instruments on IEEE ports and depending on the program, either stops the program or reinitializes the instrument and continues.

Display brightness: (rear of panel) a screwdriver slot controls brightness of characters on the display screen. Use only as much brightness as necessary.

OPERATION

Here's how to operate your controller once the program has been written and connections made to instruments and peripheral devices. (See Figure 7.)

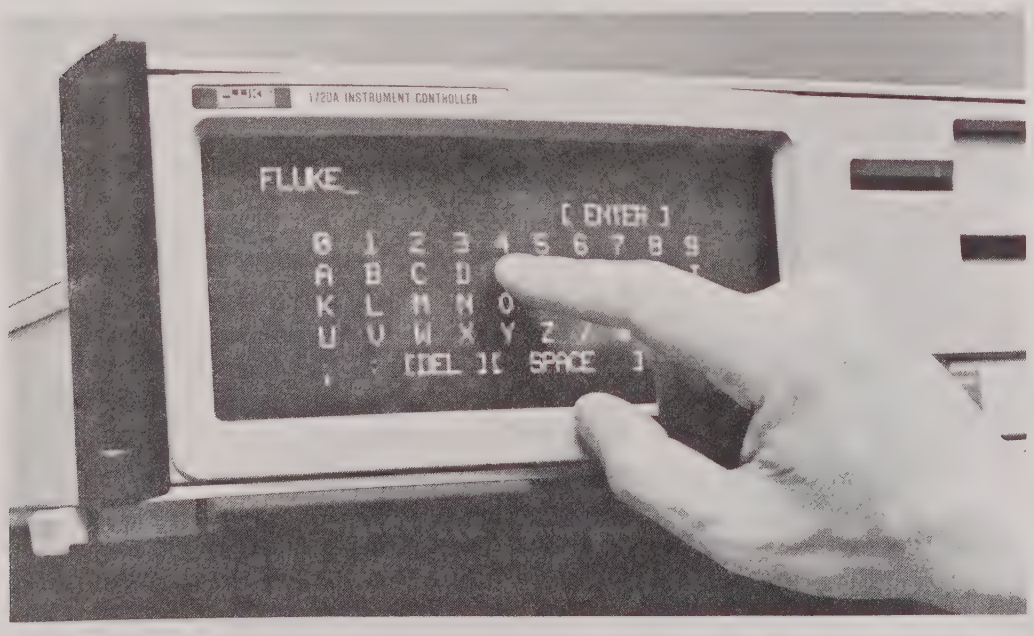


Figure 5. Touch Sensitive Display

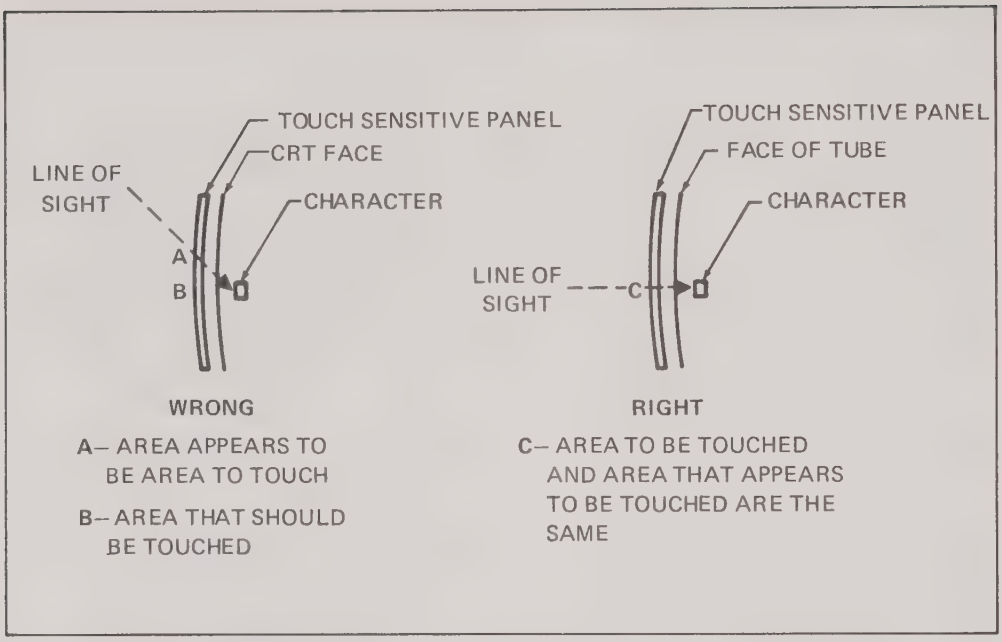


Figure 6. Parallax Distortion, Correct and Incorrect Viewing Angles

Power on Sequence (See Figure 7)

1. Insert the floppy disk. Follow the procedure outlined previously and note handling care.

2. If your 1720A is part of a system that includes a system power control, turn on system power. If system power is applied on an individual instrument basis, turn on all system instruments, including the 1720A, using the individual instrument power switches.

3. Controller runs SELF TEST once the power is turned on. (See Figure 6.) Observe any messages produced by the SELF TEST or any system test developed by your programmer. See the next section about error procedures. See your programmer for any in-house system error codes.

4. Once the SELF TEST is completed, and there are no errors, the system is ready.

NOTE

E-Disk enable switch must be in the enable mode for battery back-up.

Restart Procedure

1. Remove the floppy disk.

2. Insert a new disk as outlined in the Operating Features section and as illustrated in Figure 3.

3. Press RESTART.

4. Observe any messages which may occur.

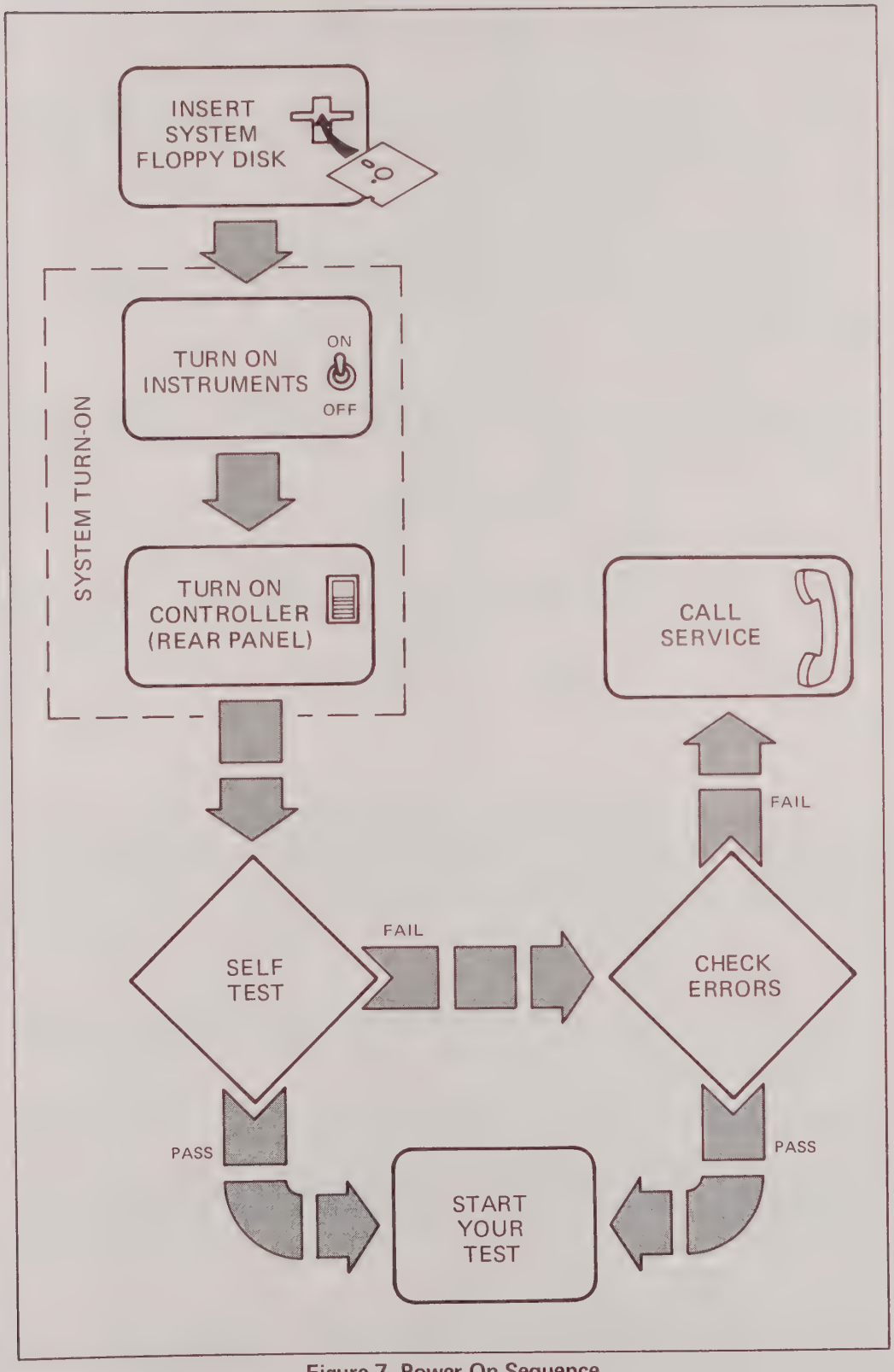


Figure 7. Power On Sequence

Power Off Sequence

1. Remove the floppy disk.

2. Turn off the system power switch or the 1720A power switch, whichever is applicable to the system you are using.

NOTE

Battery back-up for the E-Disk will come on if the E-Disk enable switch on the controller is not in the disable mode when the system power is turned off.

Error Conditions

Fluke has provided a system SELF TEST which automatically checks a number of system features at start-up. The following message is displayed on the screen while the SELF TEST is operating:

```
FLUKE 1720A CONTROLLER
HELLO
SELF TEST
SELF TEST ERRORS
```

If all components are working properly, the programs especially written for your system will start processing. The errors in Table 2 might be displayed if any module is not operating properly. Follow the procedure for action required if you observe any of the following messages:

PROGRAM PROCESSING ERRORS

After the SELF TEST is completed, the controller will start processing the program written by your programmer. Table 2 lists some common errors which may occur during program processing.

PROGRAM ERRORS

If any of the errors listed in Table 3 occur, notify your programmer. We suggest you keep a log of them for reference by a programmer. A log space is provided at the rear of the manual.

NOTE

Only the more common program errors have been presented. If you see other error indications on the display, such as, `VRBL error 901 at line 10`, *you must contact your programmer.*

Table 2. Self Test Errors

MESSAGE	ACTION
!Memory Error	Indicates a problem with the memory board. Service may be required.
!Video Error	Indicates a problem with the video board. Service may be required.
!IEEE Missing or Faulty	Indicates a problem with the IEEE Interface board. Check whether all IEEE instruments attached to the controller are on and whether all cables are securely in place. If everything seems correct and the message still displays when you depress the RESTART switch again, service may be required. You may also wish to disconnect the IEEE cables from the controller to confirm whether the instruments are the source of trouble.
!Floppy Error	Indicates a problem with the floppy disk drive or its control board.
?No System on Device	Check that the disk in the drive is a system disk. Try another system disk. You may have inserted the wrong disk or inserted it incorrectly.
?Disk not Mounted	Insert or re-insert a system disk. Either you forgot to insert a disk or it was improperly inserted. Check whether the disk is upside down or the entry latch has not been closed.
?Floppy Error	The inserted disk is faulty. Try another system disk.
?Bad Directory on Device	The directory on the disk is incorrect. Use another disk.

ERROR	ACTION
301 Write protected disk	You probably should have a data disk inserted.
306 No more room for data on E-DISK or floppy disk	Insert another disk or check with your programmer.
400 IEEE-488 Bus timing error	Check that all devices are properly connected. If message repeats, check with your programmer.
401 IEEE-488 range error	Instrument probably turned off. Check specific instrument operation.

Table 3. Common Program Error Indications

ERROR INDICATION	MEANING
0	Too much data to fit in main memory.
1	Too large a data space was reserved by the program on E-Disk or floppy disk.
2	Too much data to fit into space reserved for it on E-Disk or floppy disk.
100	System error detected by Fluke FDOS operating software system.
101	The application program on your disk is in lexical form not compatible with this version of Fluke BASIC.*
309	Data from an RS-232-C serial port either came in too fast or was ignored by the program.
310	Input from the disk or instrument was longer than 80 characters.
800	Out of data in read: an error in your application program usually caused by failure to RESTORE a DATA statement.
801	Too many data items from disk or instrument in response to an INPUT statement in your program.
802	Not enough data items from disk or instrument in response to an INPUT statement in your program.

*Programmer's Note: obtain an ASCII (.BAS) backup file to your lexical program, enter and save it lexically under the new version of Fluke BASIC.

MAINTENANCE

You can safely perform maintenance procedures. These are limited to fuse replacement and cleaning. All other maintenance requirements should be referred to qualified personnel.

Fuse Replacement

Either a 3A Slo-Blo fuse for 120V service or MDL 1½A Slo-Blo fuse for 240V service is used to protect the power line from an accidental overload. Use the following procedure to replace a fuse:

1. Turn off the instrument power.
2. Remove the power cord from the back panel.
3. Locate the fuse holder on the rear panel.
4. Replace the fuse by pressing in the fuse cap and turning it 1/8 turn counterclockwise. Use a suitable screwdriver.
5. Install the new fuse in the fuse cap and install both in the fuse holder.
6. Turn on the power.

Touch Sensitive Display

Clean the display with a soft cloth and any commercially available glass cleaner. Avoid abrasive cleaners and paper towels on the mylar surface as they may scratch it.

Air Filter

A dirty air filter may cause the system to overheat and shutdown. Use the following procedure to clean the air filter:

1. Locate the air filter on the rear panel.

2. Remove the air filter by pinching it in the center with two fingers and pulling it from the housing.

3. Clean the air filter in water and mild detergent or a mild solvent.

4. Dry the air filter thoroughly and replace it.

CAUTION

Replacing a filter which is still wet may damage internal components.

Disk Drive

The disk drive requires no operator maintenance.

CAUTION

Lubrication of the disk drive mechanism is unnecessary. Never apply oil to the disk drive mechanism. Never insert an object into the disk drive opening other than a disk, you may damage the mechanism if you do. If there is trouble with the drive, contact a Fluke Service Center.

Table 4. Fluke Technical Service Centers, U.S. and Canada

UNITED STATES

CA, Burbank
Fluke Technical Center
2020 N. Lincoln Street
Burbank, CA 91504
(213) 849-4641

CA, Santa Clara
Fluke Technical Center
2300 Walsh Avenue
Santa Clara, CA 95050
(408) 985-1200

CO, Denver
Fluke Technical Center
1980 S. Quebec Street, Unit 4
Denver, CO 80231
(303) 750-1228

FL, Orlando
Fluke Technical Center
940 N. Fern Creek Avenue
Orlando, FL 32803
(305) 896-2296

IL, Rolling Meadows
Fluke Technical Center
1400 Hicks Road
Rolling Meadows, IL 60008
(312) 398-5800

MA, Waltham
Fluke Technical Center
244 Second Avenue
Waltham, MA 02154
(617) 890-1604

MD, Rockville
Fluke Technical Center
5640 Fishers Lane
Rockville, MD 20852
(301) 770-1576

MN, Apple Valley
Fluke Technical Center
7373 West 147th Street, Suite 196
Apple Valley, MN 55124
(612) 432-9400

NC, Greensboro
Fluke Technical Center
1310 Beaman Place
Greensboro, NC 27408
(919) 273-1918

NJ, Paramus
Fluke Technical Center
West 75th Century Road
Paramus, NJ 07652
(210) 262-9550

TX, Dallas
Fluke Technical Center
14400 Midway Road
Dallas, TX 75240
(214) 233-9945

WA, Mountlake Terrace
John Fluke Mfg. Co., Inc.
21707 66th Avenue W., Suite 1
Mountlake Terrace, WA 98043
(206) 774-2206

CANADA

ALB, Calgary
Allan Crawford Assoc., Ltd.
1935 30th Avenue N.E. #14
Calgary, ALB T2E 6Z5
(403) 230-1341

ONT, Mississauga
Allan Crawford Assoc., Ltd.
6503 Northam Drive
Mississauga, ONT L4V 1J5
(416) 678-1500

QUE, St. Laurent
Allan Crawford Assoc., Ltd.
7018 Cote de Liesse
St. Laurent, QUE H4T 1E7
(514) 731-8564

Table 5. Sales and Service Locations, International

ARGENTINA
Fluke Latin American
Service Headquarters
Virrey del Pino 4071
Buenos Aries, Argentina
Tel: 523185

AUSTRALIA
Elmeasco Instrument Pty Ltd.
P.O. Box 30
Concord, N.S.W.
Australia 2137
Tel: (02) 736-2888

AUSTRIA
Walter Rekirsch
Elektronische Gerate GmbH & Co.
Vertriebs-KG.
Liechtensteinstrasse 97/6
A-1090 Vienna, Austria
Tel. 09-43-222-347646

BELGIUM
Fluke (Belgium)S.A./N.V
6, Rue de Geneve
1140 Brussels, Belgium
Tel: 09-32-2-2164090

BRAZIL
Arotec S.A
Industria e Comercio
Av Pacaembu 811
01234 Sao Paulo S.P., Brazil
Tel 826-2266

CHILE
Intronica Chile Ltda.
Casilla 16228
Santiago 9, Chile
Tel: 44940

COLOMBIA
Coasin Ltda
Carrera 13, # 37-37, Of. 407
Ap Aero 29583
Bogota DE, Colombia
Tel: 285-0230

DENMARK
Tage Olsen A/S
Ballerup Byvej 222
DK-2750 Ballerup
Denmark
Tel: (01) 2-65 81 11

ECUADOR
Proteco Coasin Cia., Ltda.
Edificia "Jerico"
Ave. 12 de Octubre #2285y
Ave. Orellana (Planta Baja)
Quito, Ecuador
Tel: 526759

EGYPT
Lotus Engineering Organisation
P.O Box 1252
Cairo, Egypt
Tel. 71617

FINLAND
Oy Findip AB
Teollisuustie 7
02700 Kauniainen
Finland
Tel: 09-358-0-502255

FRANCE
M.B. Electronique S.A.
Rue Fourny
ZAC de BUC
B.P. #31
78530 BUC, France
Tel: 09-33-1-9568131

GREECE
Hellenic Scientific
Representations Ltd.
11, Vrassida Street
Athens 615, Greece
Tel: 09-30-1-711140

HONG KONG
Gilman Industrial Bldg., 9F
107-109 Wai Yip Street
Kwun Tong
Kowloon, Hong Kong
Tel: 3-427144

Table 5. Fluke Technical Service Centers - International (cont)

INDIA
Hinditron Services Pvt. Ltd.
"Hinditron House"
412 Raj Mahal Vilas Extension
Bangalore 560 006, India
Tel: 33139

Hinditron Services Pvt. Ltd.
69 A.L. Jagmohandas Marg
Bombay 400 006, India
Tel: 365344

INDONESIA
P.T. Dwi Tunggal Jaya Sakti
Sangga Buana Bldg., 1st Floor
J1. Senen Raya 44, P.O. Box 4435
Jakarta, Indonesia
Tel: 376608

IRAN
Irantronics Company Ltd.
20 Salm Road
Roosevelt Avenue
Tehran, Iran
Tel: 828294/831564

ISRAEL
R.D.T. Electronics Engineering Ltd.
P.O. Box 75
46, Sokolov Street
Ramat Hasharon 47235
Israel
Tel. 09-972-3-483211

ITALY
Sistrel S.p.A.
Via Timavo 66
20099 Sesto S. Giovanni (Milan)
Italy
Tel: 09-39-2-2476693

Sistrel S.P.A.
Via Guiseppe Armellini #37
00143 Rome, Italy
Tel: 09-39-6-5915551

JAPAN
John Fluke Mfg. Co., Inc.
Japan Branch
1 Higashikata-machi
Midori-ku
Yokohama 226, Japan
Tel: (045) 473-5425

KOREA
Electro-Science Korea Co.
C P.O. Box 8446
Room 1201 Bowon Bldg.
490 Chongro - 5 Ka
Chongro-ku
Seoul, Korea
Tel: 261-7702

MALASIA
O'Connor's (Malaysia) SDN. BHD.
P.O. Box 91
Petaling Jaya, Selangor
West Malaysia
Tel: 51563

MEXICO
C.J. Christensen S.A. De C.V.
Melchor Ocampo 150-8
Mexico 4 D.F., Mexico
Tel: (905) 535-2258

NETHERLANDS
Fluke (Nederland) B.V.
P.O. Box 5053
5004 EB Tilburg
The Netherlands
Tel: (013) 673973

NEW ZEALAND
W & K McLean Ltd
P.O. Box 18065
Glen Innes
Auchland, New Zealand
Tel. 587-037

NORWAY
Morgenstierne & Co A/A
Konghellegate 3
P.O. Box 6688, Rodelokka
Oslo 5, Norway
Tel. 09-47-2-356110

PAKISTAN
Pak International Operations
505 Mohammadi House
1.1. Chundrigar Road
Katrachi-2 Pakistan
Tel: 221127/239052

PERU
Importaciones Y Representaciones Electronicas S.A
Avda. Franklin D. Roosevelt 105
Lima 1, Peru
Tel: 288650

PORTUGAL
Equipamentos de Laboratorio, Ltda
P.O. Box 1100
Lisbon 1002, Portugal
Tel: 09-351-1-578936/547512

REPUBLIC OF SINGAPORE
Rank O'Connor's (PTE) Ltd.
98 Pasir Panjang Road
Singapore 0511
Republic of Singapore
Tel: 637944

REPUBLIC OF SOUTH AFRICA
Fluke S.A. (Pty) Ltd.
P.O. Box 39797
Bramley 2018
Republic of South Africa
Tel: (011) 786-3170

SPAIN
Hispano Electronica S.A.
Apartado de Correos 48
Alcorcon (Madrid)
Spain
Tel: 09-3416194108

SWEDEN
Teleinstrument AB
P.O. Box 490
S-162 Vallingby 4
Sweden
Tel: 09-46-8-380370

SWITZERLAND
Traco Electronic AG
Jenatschstrasse 1
8002 Zurich
Switzerland
Tel 09-41-1-2010711

TAIWAN
CCT Associates, Inc
P.O. Box 24209
Taipei, Taiwan
Republic of China
Tel (02) 391-6894/393-5760

THAILAND
Dynamic Supply Engineering R.O.P
56 Ekamai, Sukhumvit 63
Bangkok-11, Thailand
Tel: 914434/928532

TURKEY
Erkman Electronik Aletler
Ticaret Anonim Sirketi
Necatiby Cad 92/3
Karakoy, Istanbul, Turkey
Tel. 09-44-1-5461

UNITED KINGDOM
Fluke International Corp. - UK
Colonial Way
Watford, Herts WD24TT
United Kingdom
Tel: 09-4492340511

URUGUAY
Coasin Uruguaya S.R.L.
Edifico Ciudadela
Sarandi 690 D. Esc. 109
Casilla de Correos 1400
Correo Central
Montevideo, Uruguay
Tel: 917978

VENEZUELA
Coasin C.A
Apdo. Postal 50939
Sabana Grande # 1
Caracas 105, Venezuela
Tel: 782-9109

WEST GERMANY
Fluke (Deutschland) GmbH
Krausstrasse 32
8045 Ismaning
West Germany
Tel: (089) 96251

Customers in the following countries: Bulgaria, Czechoslovakia, Hungary, Poland, Romania, U.S.S.R., and Yugoslavia, contact: Amtest Associates Ltd., P.O. Box 55, Addlestone, Surrey, KT 15 1DU, England, Tel. (0932) 52121

OPERATOR'S LOG OF PROGRAM ERRORS

Programming errors may depend on the specific situation, not necessarily the general test or procedure. When logging errors, always describe the specific stituation in which the error occurred, the units being tested and procedures involved.

This is a suggested format:

PROGRAM NAME	ERROR PRODUCING SITUATION	ERROR INDICATION

PROGRAM NAME	ERROR PRODUCING SITUATION	ERROR INDICATION

PROGRAM NAME	ERROR PRODUCING SITUATION	ERROR INDICATION

NOTE

This manual documents the Model 1720A and its assemblies at the revision levels shown in Section 4A. If your instrument contains assemblies with different revision letters, it will be necessary for you to either update or backdate this manual. Refer to the supplemental change/errata sheet for newer assemblies, or to the backdating sheet in Section 4A for older assemblies.

This manual documents the software listed below. Consult the current issue of Change/Errata for updates to this manual and the software version.

THIS MANUAL IS COMPATIBLE WITH THE DIAGNOSTIC DISK Rev. ____

1720A
Instrument Controller

Service Manual

P/N 518662
July 1980
Rev 1 1/81
Rev 2 12/82

©1982, John Fluke Mfg. Co., Inc., all rights reserved. Litho in U.S.A.

FLUKE®

WARRANTY

Notwithstanding any provision of any agreement the following warranty is exclusive:

The JOHN FLUKE MFG. CO., INC., warrants each instrument controller it manufactures to be free from defects in material and workmanship under normal use and service for the period of 90 days from date of purchase. This warranty extends only to the original purchaser. This warranty shall not apply to fuses, floppy disks or contents thereof, or any product or parts which have been subject to misuse, neglect, accident or abnormal conditions of operations.

In the event of failure of a product covered by this warranty, John Fluke Mfg. Co., Inc. will repair an instrument controller returned to an authorized Service Facility within 90 days of the original purchase; provided the warrantor's examination discloses to its satisfaction that the product was defective. The warrantor may, at its option, replace the product in lieu of repair. With regard to any instrument controller returned within 90 days of the original purchase, said repairs or replacement will be made without charge. If the failure has been caused by misuse, neglect, accident or abnormal conditions of operations, repairs will be billed at a nominal cost. In such case, an estimate will be submitted before work is started if requested.

THE FOREGOING WARRANTY IS IN LIEU OF ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS OR ADEQUACY FOR ANY PARTICULAR PURPOSE OR USE. JOHN FLUKE MFG. CO., INC., SHALL NOT BE LIABLE FOR ANY SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES, WHETHER IN CONTRACT, TORT OR OTHERWISE.

If any failure occurs, the following steps should be taken:

1. Notify the JOHN FLUKE MFG. CO., INC., or the nearest Service Facility, giving full details of the difficulty, and include the model number, type number, and the serial number. On receipt of this information, service data or shipping instructions will be forwarded to you.

2. On receipt of the shipping instructions, forward the instrument controller transportation prepaid. Repairs will be made at the Service Facility and the instrument controller returned, transportation prepaid.

SHIPPING TO MANUFACTURER FOR REPAIR OR ADJUSTMENT

All shipments of JOHN FLUKE MFG. CO., INC., instrument controllers should be made via United Parcel Service or "Best Way" prepaid. The instrument controller should be shipped in the original packing carton. If this original carton is not available a suitable container that is rigid and of adequate size may be used. However, the JOHN FLUKE MFG. CO., INC., does not recommend shipment of instrument controllers in substitute containers. In the event a substitute container must be used, the instrument controller should be wrapped in paper and surrounded with at least four inches of excelsior or other similar shock absorbing material. JOHN FLUKE MFG. CO., INC., shall assume NO risk for intransit shipment damage.

CLAIM FOR DAMAGE IN SHIPMENT TO ORIGINAL PURCHASER

The instrument controller should be thoroughly inspected immediately upon original delivery to purchaser. All material in the container should be checked against the enclosed packing list. The manufacturer will not be responsible for shortages against the packing sheet unless notified immediately. If the instrument controller is damaged in any way, a claim should be filed with the carrier immediately. (To obtain a quotation to repair shipment damage, contact the nearest Fluke Technical Center.) Final claim and negotiations with the carrier must be completed by the customer.

The JOHN FLUKE MFG. CO. INC., will be happy to answer all applications or use questions, which will enhance your use of this instrument. Please address your requests or correspondence to: JOHN FLUKE MFG. CO. INC., P.O. BOX C9090, EVERETT, WASHINGTON 98206, ATTN: Sales Dept. For European Customers: Fluke (Holland) B.V., P.O. Box 5053, 5004 EB, Tilburg, The Netherlands.*

*For European customers, Air Freight prepaid.

1720A SERVICE POLICY

WARRANTY POLICY

a. The 1720A Controller is serviced primarily through a module exchange program or by returning the unit to your local service center for repair. Diagnostic software routines have been provided that will allow the user to isolate a fault to the module level. In the event of a failure contact your local service center to arrange for a replacement module.

There is a complete support inventory at each service facility with a 24-hour turnaround time goal on exchange modules.

b. Upon request, on-site service is available for which a "transport zone fee" is charged. The rates for these charges are available from the local office of John Fluke Mfg. Co., Inc.

c. An on-site service contract may be purchased within the warranty period to assure minimum down time.

OUT-OF-WARRANTY SERVICE

a. The 1720A Controller is serviced primarily through a module exchange program or by returning the unit to your local service center for repair. Diagnostic software routines have been provided that will allow the user to isolate a fault to the module level. In the event of a failure contact your local service center to arrange for a replacement module. Module exchange will be available to the customer on a flat rate exchange basis. There will be a complete support inventory at each service facility with a 24-hour turnaround goal on exchanges.

b. On-site service is available on a time and materials basis. Additionally a transport zone fee is also chargeable.

c. An on-site service contract may be purchased to assure minimum down time. Contract support will also assure maximum priority on all service requests. Contracts initiated prior to the expiration of the warranty will not require a reinspection fee.

d. Units may be returned to any service facility for repair with quotations to the customers for estimated time and material charges.

Table of Contents

SECTION	TITLE	PAGE
1	**INTRODUCTION AND SPECIFICATIONS**	**1-1**
	1-1. DESCRIPTION OF THE 1720A DOCUMENTATION	1-1
	1-3. 1720A User Manual	1-3
	1-5. 1720A Fluke BASIC Programming Manual	1-1
	1-7. 1720A BASIC Reference Guide	1-1
	1-9. HPL to Fluke BASIC Handbook	1-1
	1-11. 1720A Display Worksheet Pads	1-1
	1-13. 1720A Operator Manual	1-1
	1-15. 1720A Service Manual	1-1
	1-17. Introduction To This Manual	1-3
	1-20. SPECIFICATIONS	1-3
2	**THEORY OF OPERATION**	**2-1**
	2-1. INTRODUCTION	2-1
	2-3. OVERALL FUNCTIONAL DESCRIPTION	2-1
	2-8. System Software	2-1
	2-10. 32K Word Main Memory	2-1
	2-12. Control of Peripheral Devices	2-3
	2-14. Data Processing	2-3
	2-16. Serial Data Capability	2-3
	2-18. Floppy Disk Storage	2-3
	2-20. Programmer Keyboard and Touch-Sensitive Display	2-3
	2-22. Bus Structure	2-3
	2-25. Optional Electronic Disk	2-3
	2-27. Interrupts	2-3
2A	**CPU MODULE**	**2A-1**
	2A-1. INTRODUCTION	2A-1
	2A-6. DECODING AND EXECUTING INSTRUCTIONS	2A-1
	2A-7. Overview	2A-1
	2A-10. Workspace Registers	2A-3
	2A-12. Internal Registers	2A-4
	2A-14. Workspace Register File	2A-4
	2A-17. Instruction Cycle	2A-4
	2A-19. MEMORY DATA TRANSFERS	2A-5
	2A-22. MEMORY MAPPED I/O OPERATIONS	2A-12
	2A-24. CRU IN/OUT DATA TRANSFERS	2A-12
	2A-28. ELECTRONIC DISK DATA TRANSFERS	2A-12
	2A-34. RESPONSE TO INTERUPTS	2A-12

(continued on page ii)

TABLE OF CONTENTS, *continued*

SECTION	TITLE	PAGE
	2A-36. Device Interrupts During RINT	2A-13
	2A-39. RESET and LOAD	2A-13
	2A-43. RESPONSE TO DMA REQUESTS	2A-13
	2A-46. EXTERNAL INSTRUCTION LOGIC	2A-15
	2A-48. CPU CLOCK AND SYSTEM SYNC	2A-15
	2A-50. TIME-OUT ERROR	2A-15
	2A-53. RS-232-C SERIAL PORTS	2A-15
	2A-58. CPU MODULE STATUS REGISTER	2A-15
2B	**MEMORY MODULE**	**2B-1**
	2B-1. INTRODUCTION	2B-1
	2B-4. MAIN MEMORY	2B-1
	2B-5. Dynamic Ram	2B-1
	2B-7. Parallel Address And Data Lines	2B-1
	2B-9. Independent Upper and Lower Address Banks	2B-1
	2B-11. 7-Bit Chip Address Bus	2B-1
	2B-13. Read and Write Operations	2B-1
	2B-17. BYTE/WORD ADDRESSING	2B-5
	2B-19. TRI-STATE BUFFERS	2B-5
	2B-21. Memory Address Buffer	2B-5
	2B-23. Data Input Buffer	2B-5
	2B-26. Data Output Buffer	2B-5
	2B-28. ON BOARD STROBE GENERATION	2B-6
	2B-30. Byte/Bank Decoder	2B-6
	2B-38. Memory Control Buffer and Timing	2B-6
	2B-41. CHIP ADDRESS MULTIPLEXING	2B-6
	2B-44. TRANSPARENT REFRESH	2B-7
	2B-47. Refresh Counter	2B-7
	2B-49. Row Column Decoder	2B-7
	2B-52. BYTE PARITY	2B-7
	2B-54. Write Parity Generation	2B-7
	2B-56. Read Parity Checking	2B-7
	2B-58. SWITCH SELECTABLE BANKS	2B-7
2C	**IEEE-488 INTERFACE MODULE**	**2C-1**
	2C-1. INTRODUCTION	2C-1
	2C-3. INTRODUCTION TO THE IEEE-488 BUS	2C-1
	2C-5. Basic Concepts and Terms	2C-1
	2C-13. Bus Operating Modes	2C-2
	2C-20. Three-Wire Handshake	2C-2
	2C-22. Operational Sequences	2C-5
	2C-26. IEEE-488 INTERFACE MODULE THEORY OF OPERATION	2C-10
	2C-27. Introduction	2C-10
	2C-30. Two Independent IEEE-488 Ports	2C-10
	2C-37. IEEE Bus Read Operation	2C-12
	2C-40. IEEE Bus Write Operations	2C-12
	2C-42. Automatic IFC	2C-14
	2C-44. Automatic DCL	2C-14
	2C-46. Five Maskable Interrupts	2C-14
	2C-51. Configuration Switches	2C-14
	2C-54. Board# Select Switch	2C-15
2D	**VIDEO/KEYBOARD**	**2D-1**
	2D-1. INTRODUCTION	2D-1
	2D-4. FEATURES OF THE VIDEO/KEYBOARD MODULE	2D-1
	2D-5. THE NORMAL/TEST SWITCH	2D-3

(continued on page iii)

TABLE OF CONTENTS, *continued*

SECTION	TITLE	PAGE
	2D-7. KEYBOARD SCANNING	2D-3
	2D-10. ASCII CODE GENERATION	2D-3
	2D-12. DISPLAY MEMORY	2D-4
	2D-14. CLICKER	2D-4
	2D-16. INTERRUPTS	2D-4
	2D-18. INTERFACE TO THE 1720A BUSES	2D-4
	2D-20. DISPLAY FORMAT	2D-5
	2D-22. CRT CONTROLLER	2D-5
	2D-24. DMA FOR DISPLAY	2D-5
	2D-29. CHARACTER GENERATION	2D-6
	2D-34. DISPLAY REFRESH AND SCROLLING	2D-7
	2D-37. CURSOR/BAUD RATE	2D-7
	2D-40. VIDEO	2D-7
	2D-48. DOUBLE SIZE CHARACTERS	2D-10
	2D-51. BEEPER	2D-10
	2D-53. LEDS	2D-10
	2D-55. CAPS LOCK LED	2D-10
	2D-57. PAGE MODE LED	2D-10
	2D-59. COMPOSITE VIDEO	2D-10
	2D-61. TEST MODE OPERATION	2D-10
2E	**DMA-FLOPPY MODULE**	**2E-1**
	2E-1. INTRODUCTION	2E-1
	2E-4. DMA CONTROLLER GENERAL DESCRIPTION	2E-1
	2E-11. Module-To-Memory DMA Transfers	2E-4
	2E-25. Memory-To-Module DMA Transfers	2E-5
	2E-39. DMA Interrupts	2E-6
	2E-43. FLOPPY DISK CONTROLLER GENERAL DESCRIPTION	2E-7
	2E-45. DMA and System Interface	2E-7
	2E-49. Commands	2E-7
	2E-60. Floppy Disk Interface	2E-9
	2E-69. Head Positioning	2E-10
	2E-71. Disk Read Operations	2E-10
	2E-75. Disk Write Operations	2E-10
2F	**POWER SUPPLY**	**2F-1**
	2F-1. INTRODUCTION	2F-1
	2F-3. SWITCHING POWER SUPPLY	2F-1
	2F-7. ±HV SUPPLY	2F-1
	2F-13. PULSE WIDTH MODULATOR AND CONTROL	2F-3
	2F-18. Switching Transistor PCB	2F-3
	2F-24. 50 kHz Power Transformer	2F-3
	2F-26. Supply Rectifiers	2F-3
	2F-28. +5V Sense and Regulation	2F-3
	2F-32. FAULT PROTECTION	2F-4
	2F-34. LOW LINE SHUTDOWN	2F-4
	2F-36. OVERCURRENT PROTECTION	2F-4
	2F-38. OVERVOLTAGE PROTECTION	2F-4
	2F-40. FAN	2F-4
	2F-42. BATTERY BACKUP	2F-4
	2F-43. Battery Backup Supply	2F-4
	2F-48. Battery Charger	2F-5
	2F-50. Low Battery Dropout	2F-5
	2F-52. STATUS LINES	2F-5
	2F-54. REMOTE CONNECTOR	2F-5

(continued on page iv)

TABLE OF CONTENTS, *continued*

SECTION	TITLE	PAGE
2G	**CLOCK MODULE**	**2G-1**
2G-1.	INTRODUCTION	2G-1
2G-2.	CRU INTERFACE	2G-1
2G-5.	Bit Assignments	2G-1
2G-7.	Clock Registers	2G-1
2G-9.	OPERATIONAL SEQUENCES	2G-1
2G-10.	Overview	2G-1
2G-12.	Setting the Time	2G-1
2G-14.	Setting the Year	2G-2
2G-16.	Checking For Valid Time	2G-3
2G-18.	Checking For Year Rollover	2G-3
2G-20.	Reading the Time	2G-3
2H	**OPTIONAL ELECTRIC DISK**	**2H-1**
2H-1.	INTRODUCTION	2H-1
2H-4.	OPERATIONAL SEQUENCES	2H-1
2H-5.	Write Operation	2H-1
2H-10.	Read Operation	2H-1
2H-15.	ADDRESS RECOGNITION	2H-2
2H-18.	MEMORY ORGANIZATION	2H-2
2H-20.	MEMORY ADDRESSING	2H-2
2H-22.	MEMORY DATA TRANSFER	2H-2
2H-28.	PARITY	2H-3
2H-30.	MEMORY REFRESH	2H-3
2H-35.	STATUS/CONTROL REGISTER	2H-5
2H-39.	BATTERY OPERATION	2H-5
2H-41.	BOARD SELECT SWITCH	2H-5
2J	**PROGRAMMER KEYBOARD, TOUCH-SENSITIVE DISPLAY**	**2J-1**
2J-1.	PROGRAMMER KEYBOARD	2J-1
2J-5.	TOUCH SENSITIVE DISPLAY	2J-1
2J-8.	CRT DISPLAY ELECTRONICS	2J-1
2K	**FLOPPY DISK DRIVE UNIT**	**2K-1**
2K-1.	INTRODUCTION	2J-1
2K-3.	THE DRIVE UNIT	2K-1
2K-8.	TRACK AND SECTOR FORMAT	2K-1
3	**MAINTENANCE**	**3-1**
3-1.	INTRODUCTION	3-1
3-5.	REQUIRED EQUIPMENT	3-1
3-7.	LIVE BOX PROCEDURE	3-1
3-8.	Introduction	3-1
3-12.	Live Box Procedure Steps	3-2
3-15.	DEAD BOX PROCEDURE	3-5
3-16.	Introduction	3-5
3-19.	Dead Box Procedure Steps	3-7
3-20.	ASSEMBLY DISASSEMBLY PROCEDURES	3-19
3-21.	Introduction	3-19
3-23.	Tools Required	3-19
3-26.	Module (PCB)	3-19
3-28.	Power Supply	3-19
3-30.	Battery	3-20

(continued on page v)

TABLE OF CONTENTS, *continued*

SECTION	TITLE	PAGE
	3-32. Floppy Disk Drive	3-20
	3-34. CRT Assembly	3-21
	3-36. Touch-Sensitive Display	3-23
	3-38. CRT Electronics Assembly/Disassembly Procedure	3-25
	3-41. Battery Fuse	3-27
	3-43. HDT and Standard Prom Sets	3-27
	3-48. Programmer Keyboard PCB	3-28
	3-50. Video Display Alignment Procedure	3-28
	3-53. PREVENTIVE MAINTENANCE PROGRAM	3-30
4	**GENERAL INFORMATION**	**4-1**
4A	**MANUAL CHANGE INFORMATION**	**4A-1**
5	**LIST OF REPLACEMENT PARTS**	**5**
	TABLE OF CONTENTS	5-1
	5-1. INTRODUCTION	5-2
	5-4. HOW TO OBTAIN PARTS	5-2
6	**SCHEMATICS**	**6-1**
	TABLE OF CONTENTS	6-1

List of Tables

TABLE	TITLE	PAGE
1-1.	1720A Specifications	1-2
2-1.	Bus Signals	2-5
2-2.	Bus Signal Definitions	2-5
2A-1.	Electronic Disk Register Assignments	2A-13
2A-2.	External Instructions	2A-16
2A-3.	Clock and Timing Outputs	2A-16
2A-4.	RS-232 Port Signal Definitions	2A-16
2A-5.	CPU Status Register	2A-16
2B-1.	Strobe Mnemonics	2B-4
2B-2.	Byte/Word Addressing and Control	2B-5
2B-3.	Data Input Buffer Mode Control	2B-5
2B-4.	Data Output Buffer Control	2B-5
2C-1.	IEEE-Interface Function Repertoire	2C-2
2C-2.	IEEE Bus Signal Lines	2C-3
2C-3.	Command Message Categories	2C-5
2C-4.	Command Bus Messages	2C-6
2C-5.	IEEE Interface Register Address Decoding	2C-13
2E-1.	Register Address Assignments	2E-3
2E-2.	DMA Controller Control Register	2E-3
2E-3.	Status Register	2E-4
2E-4.	Command Summary	2E-7
2E-5.	Flag Summary	2E-8
2G-1.	Bit Assignments	2G-2
2G-2.	Clock Register Assignments	2G-2
2H-1.	Electronic Disk Address Assignments	2H-1
2H-2.	Register Assignments	2H-1
2H-3.	Electronice Disk Board Select Switch	2H-5
3-1.	Required Equipment	3-2
3-2.	Interpreting CVFY Results	3-4

List of Illustrations

FIGURE	TITLE	PAGE
	Frontispiece 1720A Instrument Controller	*x*
1-1.	Manual Set	1-2
2-1.	1720A System Configuration	2-2
2-2.	1720A Controller Internal Buses	2-4
2A-1.	CPU Environment	2A-1
2A-2.	CPU Module Block Diagram	2A-2
2A-3.	Memory Map	2A-3
2A-4.	Workspace Registers	2A-4
2A-5.	Locating A Specific Resister	2A-5
2A-6.	Microprocessor Flow Chart	2A-6
2A-7.	Microprocessor Pin Assignments	2A-7
2A-8.	Memory Timing	2A-10
2A-9.	CPU Address Bus	2A-11
2A-10.	Memory Area Selection	2A-11
2A-11.	CRU Interface Timing	2A-14
2B-1.	Memory Module Block Diagram	2B-2
2B-3.	Main Memory Chip Layout	2B-4
2C-1.	IEEE Bus Connections	2C-4
2C-2.	Handshake Flowchart	2C-9
2C-3.	Handshake Timing	2C-9
2C-4.	Two Possible Minimum Systems	2C-10
2C-5.	IEEE Module Block System	2C-11
2C-6.	Dual IEEE-488 Module Configuration Switches	2C-15
2D-1.	Video/Keyboard Interface Block	2D-2
2D-2.	Display Memory	2D-4
2D-3.	Character Cell	2D-6
2D-4.	Display of Character Row	2D-8
2D-5.	CRT Video Signals	2D-9
2E-1.	Floppy Block Diagram	2E-2
2F-1.	Power Supply Block Diagram	2F-2
2H-1.	Electronic Disk Memory Organization	2H-3
2H-2.	Electronic Disk Block Diagram	2H-4
2J-1.	Touch Sensitive Display	2J-2
2J-2.	CRT Electronics Block Diagram	2J-2
2K-1.	Track and Sector Format	2K-2
3-1.	Sample CVFY Summary	3-3
3-2.	Floppy Disk Drive PCB	3-6
3-3.	Cat's Eye Alignment Waveform	3-7
3-4.	Power Supply Verification	3-8
3-5.	CRT Alignment Pattern	3-29

1720A Instrument Controller

Section 1
Introduction and Specifications

1-1. DESCRIPTION OF THE 1720A DOCUMENTATION

1-2. Several documents are available to serve the needs of a variety of users. Figure 1-1 illustrates the manual set. To obtain additional copies of these manuals, or others when developed, check with a Fluke Sales Office, listed at the end of this manual.

1-3. 1720A User Manual

1-4. The User Manual is an introductory manual for the programmer or system designer who is using the 1720A to set up an instrumentation system. It presents both the purpose of and the interaction between the various software and hardware resources in the 1720A. In addition, it serves as a reference guide for the Console Monitor, the File Utility Program, and the other utility programs. Order Fluke Part Number 518654.

1-5. 1720A Fluke BASIC Programming Manual

1-6. The Programming Manual provides a description of the Fluke-Enhanced ANSI-standard BASIC language developed for the 1720A Instrument Controller. It is arranged by subject, and combines readable syntax diagrams with numerous examples. Emphasis is on instrumentation system control using effective programming techniques. This manual presumes familiarity with the 1720A User Manual. Order Fluke Part Number 518670.

1-7. 1720A BASIC Reference Guide

1-8. The BASIC Reference Guide is a pocket-size quick reference guide that summarizes the most often used contents of the User and BASIC Programming manuals. Order Fluke Part Number 526210.

1-9. HPL to Fluke BASIC Handbook

1-10. A specialized manual for the programmer who is familiar with the Hewlett-Packard 9825 Calculator using the HPL language. Building upon this familiarity, the handbook describes functional similarities and differences, and leads the programmer through the task of converting existing HPL programs for operation on the 1720A Controller. Order Fluke Part Number 546341.

1-11. 1720A Display Worksheet Pads

1-12. The 1720A display worksheets have a grid pattern that shows both normal and double size character positions, and the touch sensitive areas of the display. This allows the programmer to design effective display layouts. They are available in pads of 50. Order Fluke Part Number 533547.

1-13. 1720A Operator Manual

1-14. The Operator Manual is a brief manual for the operator of a programmed 1720A-controlled system who will not normally use the keyboard. It discusses such subjects as proper disk handling and error interpretation, and includes an error and trouble-indicent log. Order Fluke Part Number 518647.

1-15. 1720A Service Manual

1-16. The Service Manual includes component level theory of operation for each hardware module, with diagnostic procedures that can resolve failures to the modular level. Schematics and parts lists are provided. Order Fluke Part Number 518662.

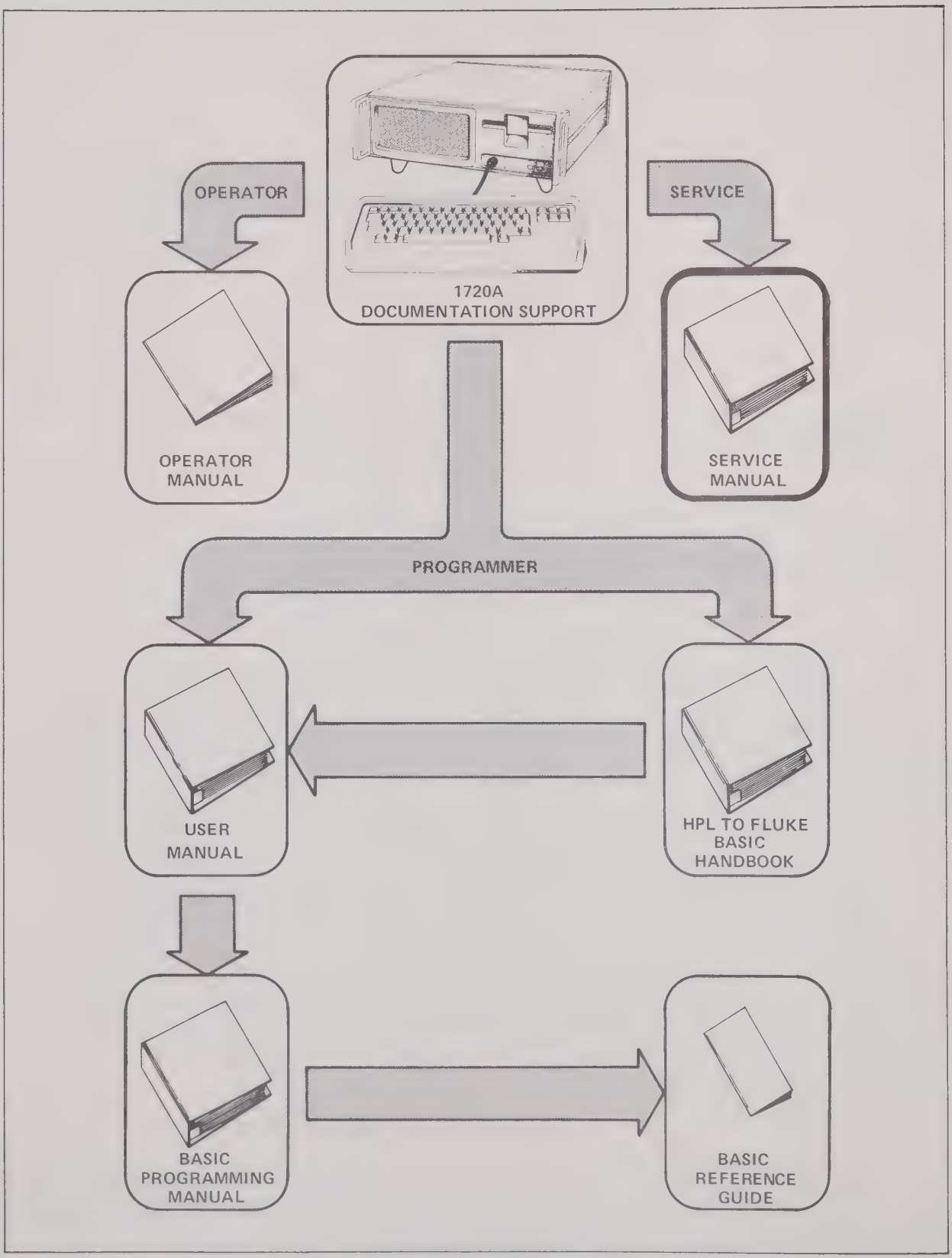


Figure 1-1. Manual Set

1-17. INTRODUCTION TO THIS MANUAL

1-18. The 1720A Service Manual is intended to be a reference source for the Customer Service Engineer. The component-level theory of operation is divided into a series of subsections. The first of these sections, Section 2, is an overview of the entire system. This section presents an overall functional description of the 1720A. Each of the subsections that follows Section 2 gives a detailed theory of operation for a particular system module or functional group of modules. Each of these subsections has an introduction that ties its detailed theory of operation into the overview in Section 2.

1-19. The Maintenance Section, Section 3, begins with a performance verification procedure that allows the user to verify that the system is performing correctly. This performance verification procedure is followed by a modular level diagnostic procedure and assembly/disassembly procedures that can resolve and enable repair of failures to the modular level. The last two sections of the manual contain the parts lists and schematics.

1-20. SPECIFICATIONS

1-21. Refer to Table 1-1 for the 1720A Instrument Controller specifications. In addition to these specifications, the 1720A has the following features:

- Touch-sensitive display for operator which serves as a dynamic, tutorial, programmer-defined "keyboard"
- Detachable programmer's keyboard
- Rackmountable
- Two character sizes, programmable
- 128K or 256K byte E Disk (electronic disk) optional (K = 1024)
- 60K bytes of read/write memory with byte parity 175K-byte mini-floppy disks for archives and back-up
- Real time and calendar clock
- Dual IEEE-488 interfaces, standard
- Dual RS-232-C interfaces, standard
- Composite video output, standard

Table 1-1. 1720A Specifications

GENERAL SPECIFICATIONS

Operating Temperature	10°C to 35°C
Storage Temperature	-10°C to +30°C for 1720A, +10°C to +52°C for floppy disks
Relative Humidity (Operating)	8% to 80%
Relative Humidity (Non-operating)	5% to 95%, (8% to 90% for floppy disks), noncondensing
Size	14.6 cm H x 43.18 cm W x 62.24 cm L including feet and handles (5.25 in H x 17 in W x 20.5 in L)
Weight	Controller 16 kg (35 lbs.); Keyboard 1.4 kg (3 lbs.)
Power	90V to 132V at 47 Hz to 63 Hz, 192V to 264V at 47 Hz to 63 Hz, 104V to 126V at 380 Hz to 420 Hz

CRT DISPLAY SPECIFICATIONS

Scanning Method	Non-interlaced raster scan
Refresh Rate	60 Hz at all times
Refresh Memory	1280 bytes of dedicated display memory: 16 lines of 80 characters each.
CRT Screen	High contrast green phosphor, low profile, rectangular
Screen Capacity	16 x 80 cells, or 8 x 40 cells
Standard Character Set	128 ASCII characters
Character Font	7 x 9 in an 8 x 14 matrix
Highlighting Capability	Inverse video, blinking, underline, and highlighting
Cursor	Blinking underline standard
Alpha Raster Size	7.6 cm x 19 cm (3.0 in x 7.5 in)
Screen Brightness	Adjustable with rear control

Table 1-1. 1720A Specifications (cont)

Display Alignment	Character with respect to touch-sensitive grid at 25°C: ±1 character horizontal ±0.75 character vertical Change in character positions over the Operating Temperature Range: ≤1.5 character horizontal ≤0.3 character vertical

Section 2
Theory of Operation

2-1. INTRODUCTION

2-2. This section presents an overall functional description that is intended to give the general concept of what the Controller is and how the modules within it work together. The detailed theory that follows introduces each of the system modules by tying the functions each module performs into the overall functional description. This is followed by a description of how the circuits within that module implement each of those functions. By considering a system malfunction in terms of symptoms that relate to these functions, the Customer Service Engineer can go directly to the pertinent description.

2-3. OVERALL FUNCTIONAL DESCRIPTION

2-4. The Model 1720A is designed for use as a programmable Instrument Controller in a multiple instrument system using the IEEE-488 Bus. In the capacity of controller, it is capable of sending universal commands and addressed commands and addressing devices on the bus to designate talkers and listeners. The user programs are written in BASIC and stored in and executed from a 60K byte Memory module. They give the 1720A the ability to process collected data, format the results to meet unique user needs, make decisions based on events, and respond to instrument service requests with user specified routines.

2-5. In addition to this main function, the 1720A stores programs and data on a built-in floppy disk and provides two general purpose RS-232-C I/O Ports which can interface to many devices including printers and modems. The 1720A can operate up to two optional 128K byte Electronic Disk Modules which provide electronic disk-like storage.

2-6. The programmer enters programs via a detachable keyboard. The system software communicates with the programmer via a CRT display. After entering these programs, the programmer can store them on floppy disks or the Electronic Disk and load them from disks from then on.

2-7. The operator operates the system with the keyboard detached by using the Touch-Sensitive Display. This permits the operator to respond to directions from the program which appear on the CRT display and to give inputs to the system by touching specific points on the CRT face.

2-8. System Software

2-9. The 1720A System Configuration is shown in Figure 2-1. The system software consists of the FDOS (Floppy Disk Operating System), the Console Monitor, the BASIC Interpreter, a ROM Loader, plus the File Utility or other utility program. The user can write programs in BASIC which are stored in, and executed from the Memory module. These programs enable the 1720A to function as a programmable instrument controller with data processing capabilities. FDOS allows programs and data to be stored on the built-in floppy disk. Consult the User Manual for a complete discussion of these capabilities.

2-10. 32K Word Main Memory

2-11. The main memory, located in the Memory Module, has a 32K word capacity. These words are accessed at even memory addresses by the CPU Module; however, 2K words of the main memory cannot be accessed by the user. This 2K of address space accesses 2K of memory in the CPU Module itself called scratch pad RAM. The discussion on the CPU Module covers this overlay of address space in detail. Each word in the main memory consists of two 8-bit bytes giving the memory a user capacity of 60K bytes. Bytes are treated by the system as odd and even bytes during DMA (Direct Memory Access) operations by the Floppy Disk Module. During these data transfers, the odd byte is accessed with an odd address.

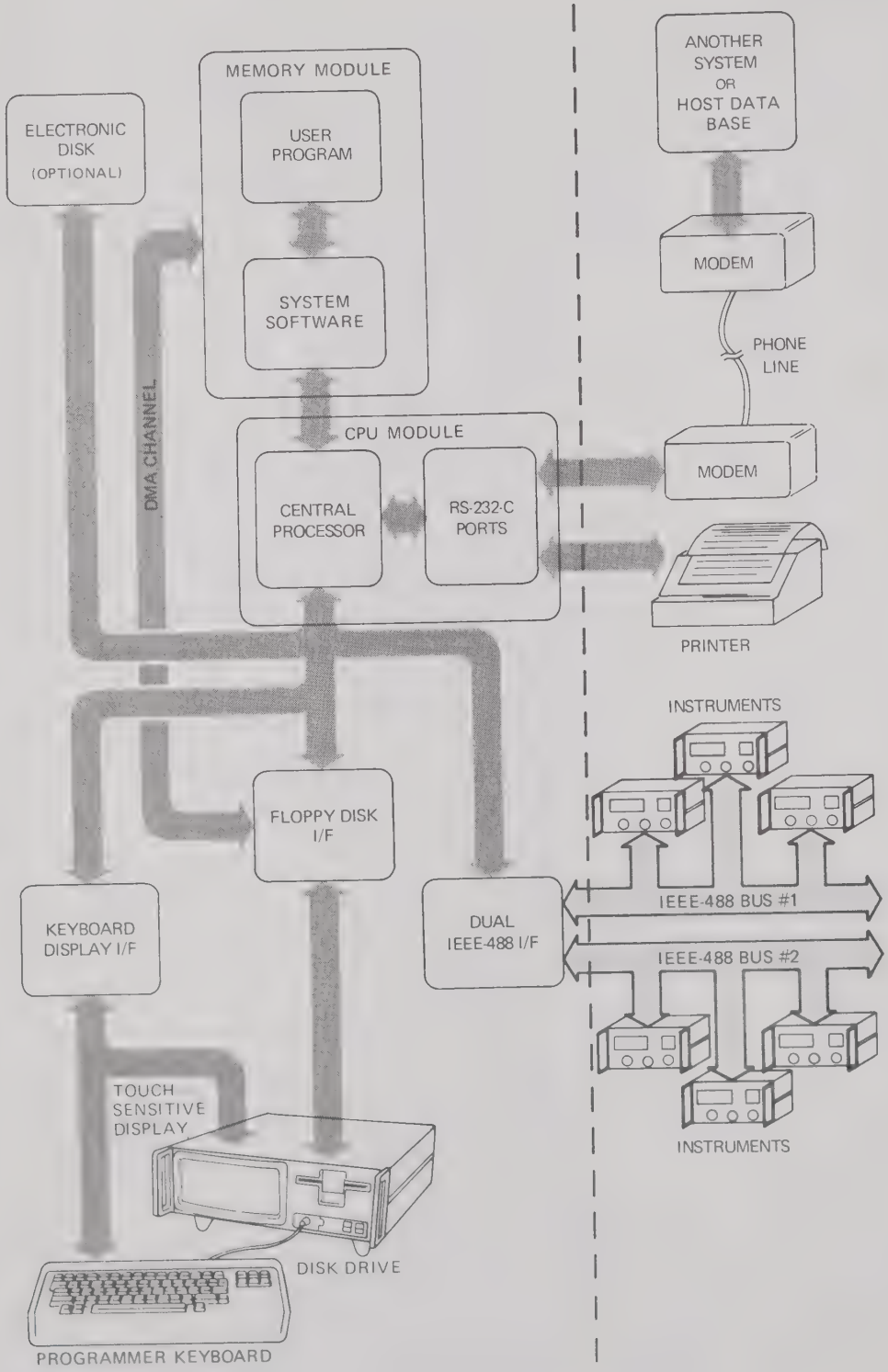


Figure 2-1. 1720A System Configuration

2-12. Control of Peripheral Devices

2-13. As shown in Figure 2-1, the 1720A can control external instruments via two independent IEEE-488-1978 buses. Up to 15 devices can be controlled on each IEEE bus (including the Controller). It is through these two interfaces that the 1720A carries out its primary function of instrument control under the direction of the user program.

2-14. Data Processing

2-15. The 1720A contains a CPU Module that has a Central Processor which incorporates a complete mini computer instruction set. This module can process the data which the 1720A receives from the external devices and from the system operator. A full description of the capabilities available to the user/programmer can be found in the User Manual.

2-16. Serial Data Capability

2-17. The 1720A has two RS-232-C ports. The interface for these ports is located on the CPU Module. As shown in Figure 2-1, these ports can interface the 1720A to many kinds of devices including a printer for direct data recording, a modem for data transfer, another system, or a host data base.

2-18. Floppy Disk Storage

2-19. The Floppy Disk Interface Module drives the built in floppy disk drive. This provides 175K bytes of user storage on a 5-1/4-inch double-density diskette. The Floppy Disk Module uses DMA (Direct Memory Access) data transfers for rapid storage and retrieval of data.

2-20. Programmer Keyboard and Touch-Sensitive Display

2-21. Both the Programmer Keyboard and the Touch-Sensitive Display pass data to and from the rest of the system via the Video/Keyboard Module. This Module communicates directly with the CPU Module via a unique serial interface called the CRU (Communications Register Unit) Bus.

2-22. Bus Structure

2-23. The 1720A is designed around four internal buses, as shown in Figure 2-2. They are:

1. Address Bus
2. Data Bus
3. CRU Bus
4. Control Bus

2-24. Table 2-1 provides a description of the bus signals. It lists the name, mnemonic, number of lines, driver type, and active level for every bus signal. Table 2-2 defines these signals. Refer to these two tables while reading this manual. Familiarity with the signals on these four busses is vital to the understanding of the 1720A.

2-25. Optional Electronic Disk

2-26. The optional Electronic Disk Module or E-Disk* has 128K bytes of memory capacity. Up to two Modules can be installed and the 1720A treats the pair as one large electronic disk. This gives a total capacity of 256K bytes. Unlike the Memory Module, each address in the Electronic Disk (both odd and even) contains one word. This leaves an Electronic Disk with its 128K byte capacity accessible by word only. In addition to this difference, the CPU Module accesses the Electronic Disk via Memory Mapped I/O operations instead of simple memory read/write operations.

2-27. Interrupts

2-28. The theory of operation for the CPU Module includes a detailed discussion on the handling of interrupts. This discussion only presents a brief introduction to the use of interrupts by the 1720A as an interrupt driven system. The term interrupt is defined as an attention signal sent to the CPU by another module to obtain service.

2-29. An interrupt occurs for a variety of reasons. For example, the depression of a key on the Programmer Keyboard. When this happens, the Video/Keyboard Module needs to send the code of this key to the CPU Module. After the CPU Module asserts (brings low) the RINT- (Refresh/Interrupt) control line, the Video/Keyboard Module pulls its assigned data line low.

2-30. The CPU carries out a programmed response to each of the interrupts on the data bus according to their priority; data bit O (DIOO) has the highest priority, and data bit 15 (DI15) has the lowest. If the priority of the interrupt is higher than the one the CPU is servicing at the time of the interrupt, the CPU suspends processing of that interrupt and services the new interrupt. When done, it returns to process the original interrupt, provided no new interrupts of a higher priority occur. This nesting of interrupts can go on indefinitely.

* E-Disk is a trademark of the John Fluke Mfg. Co., Inc.

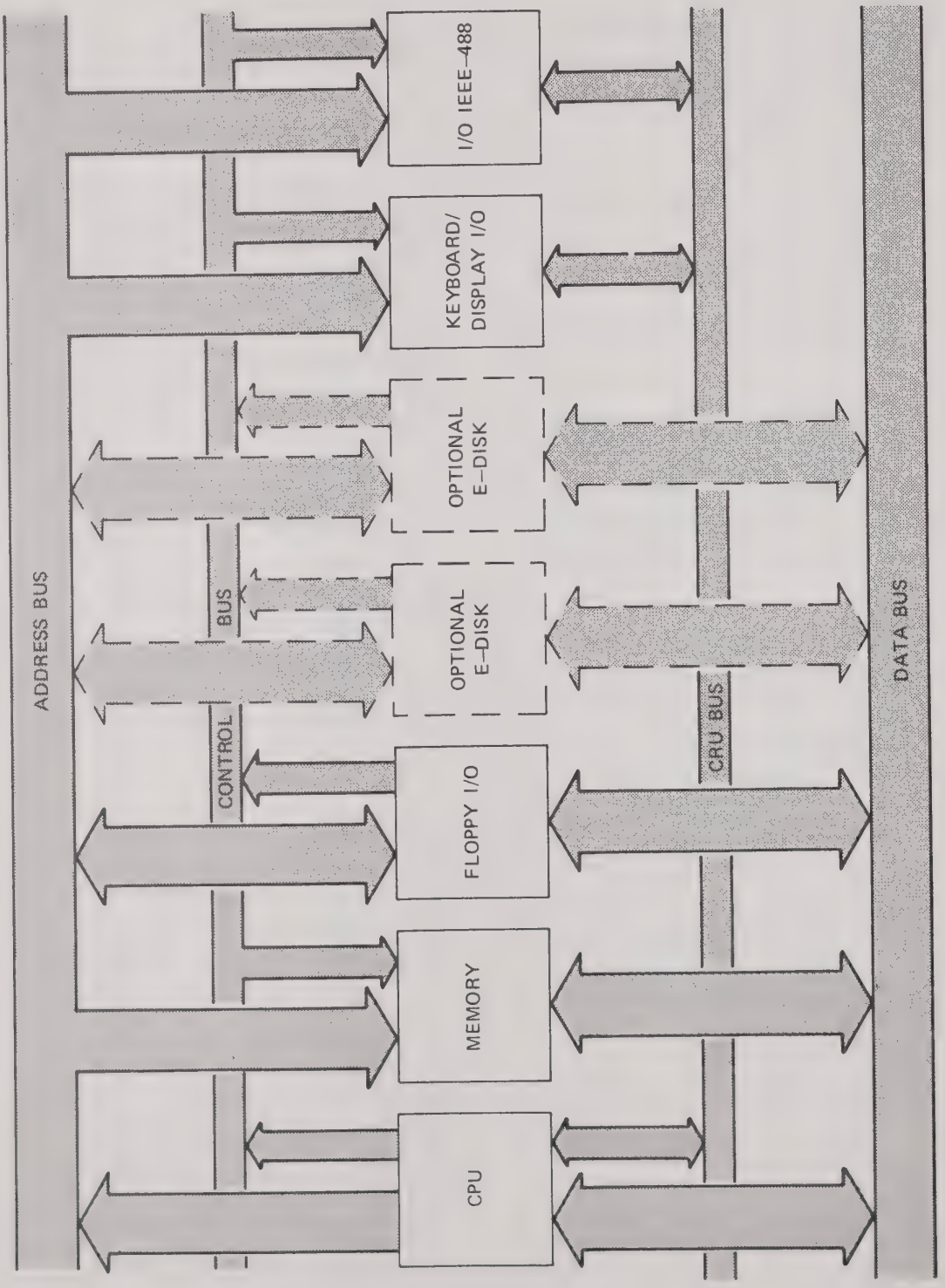


Figure 2-2. 1720A Controller Internal Buses

Table 2-1. Bus Signals

NAME	MNEMONIC	NUMBER OF LINES	TYPE OF DRIVER	ACTIVE LEVEL
Address Bus	A00-A19	20	Tri-State Buffer	High
Data Bus	DI00-DI15	16	Tri-State Buffer	High
CRU Bus				
CRU Input	CRIN	1	Open Collector	High
CRU Output	CROUT	1	TTL Buffer	High
CRU Clock	CRCLK-	1	TTL Buffer	Low
CRU Address Acknowledge	CRACK-	1	Open Collector	Low
Control Bus				
Address Valid	ACVAL-	1	Tri-State Buffer	Low
Address Acknowledge	ADACK-	1	Open Collector	Low
Bus Write	BUSWR-	1	Tri-State Buffer	Low
Refresh-Interrupt	RINT-	1	Tri-State Buffer	Low
Bus Request In	RQIN-	1	TTL	Low
Bus Request Out	RQOUT	1	TTL	Low
Bus Grant In	GRIN	1	TTL	Low
Bus Grant Out	GROUT-	1	TTL	Low
Clock Sync	SYNC	1	TTL Buffer	N/A
Byte Mode	BMODE	1	TTL	Low
Hardware Error	HERR-	1	Open Collector	Low
Bus Clear	BCLR-	1	TTL Buffer	Low
ACOK	ACOK	1	TTL Buffer	High
DCOK	DCOK	1	TTL Buffer	High
Test	TEST-	1	TTL Buffer	Low
Halt	HALT-	1	Open Collector	Low
Power Supply Lines				
+5 Volts	+5V	2		
+12 Volts	+12V	2		
Logic Common	LCOM	8		
-12 Volts	-12V	1		
Battery Backed Up Supplies				
+5 Volts	+5VB	1		
-5 Volts	-5VB	1		
+12 Volts	+12VB	1		

Table 2-2. Bus Signal Definitions

SIGNAL NAME	DEFINITION
ADDRESS BUS	(A00-A19) - 20 lines used by the processor and Direct memory Access (DMA) devices to address a word of memory or an I/0 Register. The lower 12 bits (A00-A11) are also used for Communications Register Unit (CRU) addressing.
DATA BUS	(DI00-DI15) - 16 lines for data transfer throughout the system. In addition, when the Refresh-Interrupt control line is asserted, a device requests interrupt service by driving its assigned data bus line to the low (logic 0) state. Interrupts are prioritized with a request on data bit 0 (DI00) having the highest priority and data bit 15 (DI14) the lowest. Only one interrupting device is allowed per line.
ADDRESS VALID	(ADVAL-) - Indicates the presence of valid address information on the address bus. Also indicates that data on the DATA/INTERRUPT lines is true during processor write operations.

Table 2-2. Bus Signal Definitions (cont)

SIGNAL NAME	DEFINITION
ADDRESS ACKNOWLEDGE	(ADACK-) - Response line from an addressed memory or I/0 device. Must be asserted to indicate address recognition and allow processor (or DMA device) to procees with I/0 or memory operation.
BUS WRITE	(BUSWR-) - Indicates direction of data transfer on DATA/INTERRUPT lines. A low level indicates that processor (or DMA device) is sending data to the address I/0 or memory register.
REFRESH-INTERRUPT	(RRINT-) When asserted, indicates that interrupting devices may request interrupt service by driving the appropriate data line low. Dynamic memory devices use this line to initiate a refresh cycle.
BUS REQUEST IN	(RQIN-) - Active low request by device which wishes to control the bys (typically a high speed I/0 device). This line changes at each connector. It is called Bus Request Out (RQOUT-) as it leaves the connector, having passed through the device installed in that connector. From this connector, the line goes to the next higher priority device (closer to processor) where it will again be called Bus Request In (the processor has only a RQIN- line). The device which is physically closest to the processor has the highest request priority. Each device inhibits transfer of the RQIN signal to the RQOUT line if and only if it is also requesting the bus and no lower priority device has been granted the bus, signified by a low logic level on Bus Grant In (GRIN-).
BUS REQUEST OUT	(RQOUT-) - Active low signal to signify that a device is requesting control of the bus. This line will be asserted by the device to request bus usage or if the Bus Request In line is asserted. A device may not request the use of the bys if it has already been granted to a lower priority device signified by a low level on Bus Grant In. See also Bus Request In. A non-DMA device should connect Bus Request In to Bus Request Out to maintain continuity through the device.
BUS GRANT IN	(GRIN-)-Active low indication that the processor has places its tri-state bus drivers in the high impedance mode and entered the hold state. This line is broken at each connector and is called Bus Grant Out (GROUT-) as it leaves each device enroute to the next lower priority devie (further from processor), where it is called Bus Grant In.
BUS GRANT OUT	(GROUT-) - Each device normally transfers the logic state of the Bus Grant In line to the Bus Grant Out line. A device inhibits this transfer only when it is requesting or has been granted control of the bus. At this time, the Bus Grant Out line is maintained at a logic high level. Non-DMA devices shouldconnect Bus Grant In to Bus Grant Out to maintain continuity of the grant signal.
CLOCK SYNC	(SYNC) - Six megahertz clock.
BUS CRU INPUT	(CRIN) - communications Register Unit (CRU) input line, sampled during execution of the Store Communications Register (STCR) and Test Bit (TB) instructions.
BUS CRU OUTPUT	(CROUT) - CRU output line. Serial data is placed on this line during execution of the Load Communication Register (LDCR), Set Bit to Zero (SBZ), and Set Bit to One (SBO) instructions. Data is valid when the CRU clock line is low.
CRU ADDRESS ACKNOWLEDGE	(CRCLK-) - Active low indication that valid data appears on the CROUT line. (CRACK-) - Active low response line from addressed CRU devices. Must be asserted by CRU device when its address is placed on the address bus.
HARDWARE ERROR	(HERR-) - Active low line indicates that a device has detected a fatal error, such as a memory parity error, during a read or refresh operation. Assertion of this line will initiate a processor load sequence immediately following the instruction being executed.
BUS CLEAR	(BCLR-) - A low level on this line indicates that the bus interface circuitry on all devices should be reset to the unaddressed and inactive state. BDLR- is active for 160 ns.
ACOK	(AC OK) - A high level indicates that the unit is receiving adequate ac power. A transition from the high to low state indicates that the unit may soon lose dc power and backup operations should commence.

Table 2-2. Bus Signal Definitions (cont)

SIGNAL NAME	DEFINITION
BYTE MODE	(BMODE) - A low level indicates that the bus master is expecting data in byte format. DI08 through DI15 contain valid data. DI00 through DI07 are not used.
DCOK	(DC OK) - This line is intended for use as a power on/off reset. When this line as low all operation in the unit umst cease since the power supplies are close to falling out of regulation. The only exception will be circuitry capable of rumming off the battery supply (BATT).
TEST	(TEST-) - This signal enables those devices with special diagnostic circuits to enter a special mode of operation.
HALT	(HALT-) - This active low line will initiate a load sequence.

Section 2A
CPU Module

2A-I. INTRODUCTION

2A-2. The CPU Module gives the 1720A Controller the ability to accept user programs and carry out the intent of those programs. As shown in Figure 2A-1, the 1720A user works with the CPU Module, a piece of hardware, through three pieces of software: the User Program, the BASIC Interpreter, and the FDOS (Floppy Disk Operating System). This software resides in the 60K bytes of RAM in the Memory Module. The main function of the CPU Module is to decode and execute the various instructions that make up the software.

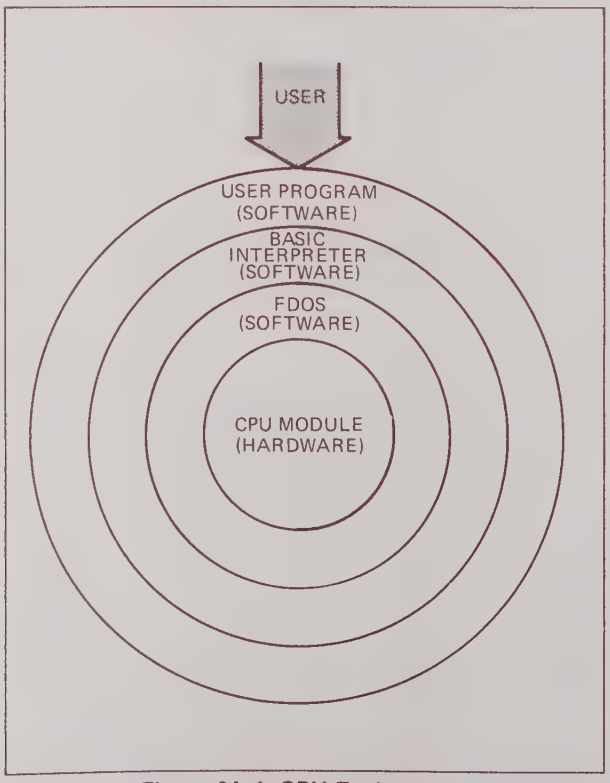


Figure 2A-1. CPU Environment

2A-3. It is important to know that the instructions the CPU receives from memory are in a special binary coded form called machine language and are part of the instruction set for the microprocessor. Refer to Figure 2A-2 and note that the microprocessor block is the main component of the CPU Module. This block is responsible for executing the user program and directing the activities of the CPU Module.

2A-4. In addition to decoding and executing instructions within the microprocessor instruction set, the CPU Module carries out other tasks. These tasks are discussed in the following paragraphs.

2A-5. Throughout the following discussion, the conventions listed below apply:

1. The same functional blocks appear on both the CPU schematic and the CPU Module block diagram.
2. Hexadecimal numbers indicate various boundries in memory or the bit patterns for codes.
3. The terms "vector" or "trap vector" refer to an address that points to the beginning of a program (including firmware routines) designed to handle a specific event, usually an interrupt. The adjective "trap" implies that a specific set of conditions leads to the vector.

2A-6. DECODING AND EXECUTING INSTRUCTIONS

2A-7. Overview

2A-8. The microprocessor receives the instructions it executes on the Internal Data Bus. This Bus receives data from three sources:

1. ROM
2. Scratch pad RAM
3. 1720A Data Bus (Memory Module)

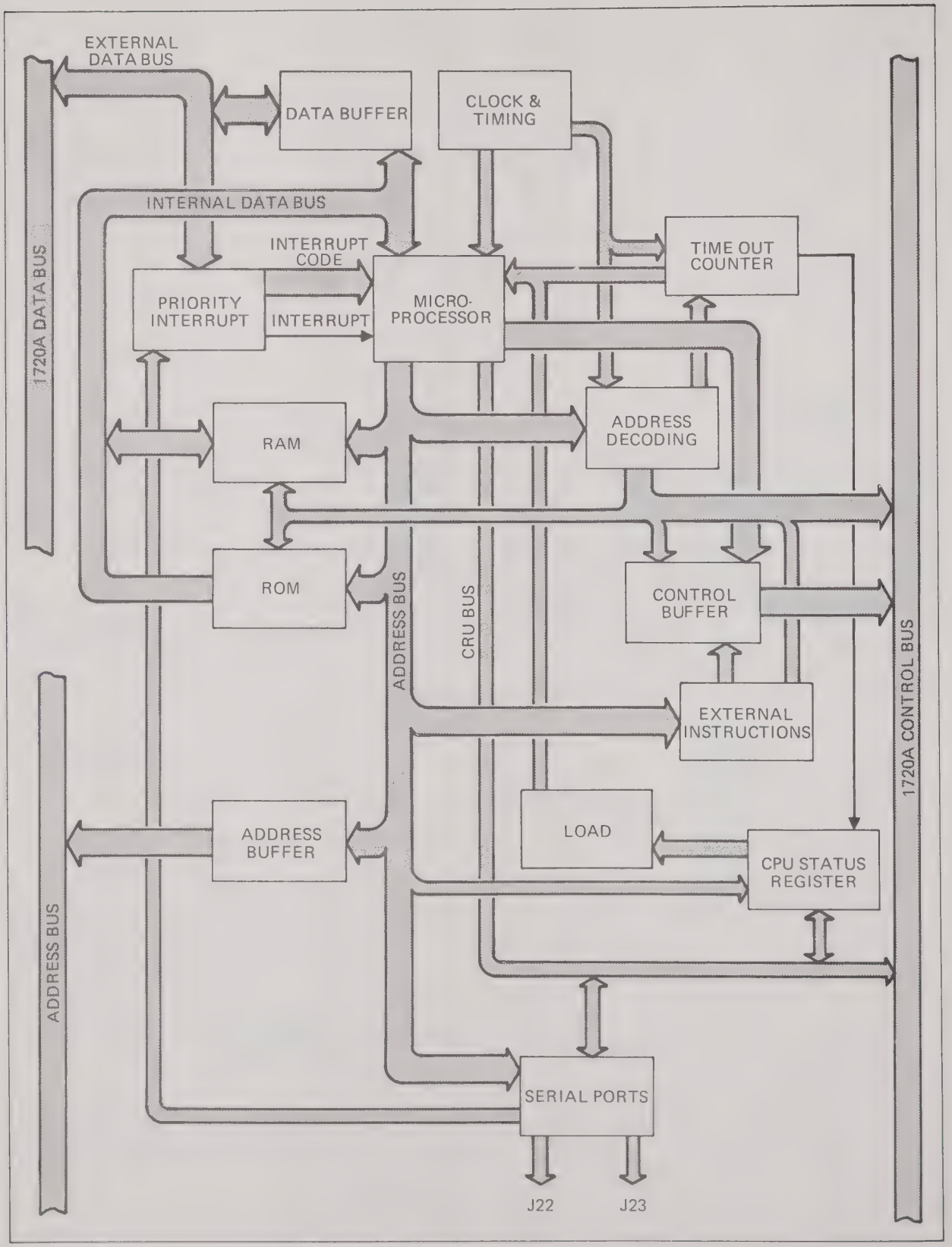


Figure 2A-2. CPU Module Block Diagram

2A-9. The actual transfer of both data and instructions to and from memory is covered later in this section. This topic covers only the way the microprocessor decodes and executes the instructions it receives. Each instruction performs one of the following operations:

1. Arithmetic, logic, compare, or bit/byte manipulation operations on data.
2. Loading or storage of internal registers (Program Counter, Workspace Pointer, or Status).
3. Serial data transfer between memory and external devices.
4. Control Functions, such as the External Instructions.

2A-10. Workspace Registers

2A-11. The microprocessor employs memory-to-memory architecture. Blocks of memory designated as workspaces replace conventional internal-hardware registers. Figure 2A-3 is a memory map that shows how the microprocessor views memory. The first 32 words are used for interrupt trap vectors. The next contiguous block of 32 memory words is used by the extended operation (XOP) instruction for trap vectors. The remaining memory is available for programs, data, and workspace registers. The special areas are available for general memory.

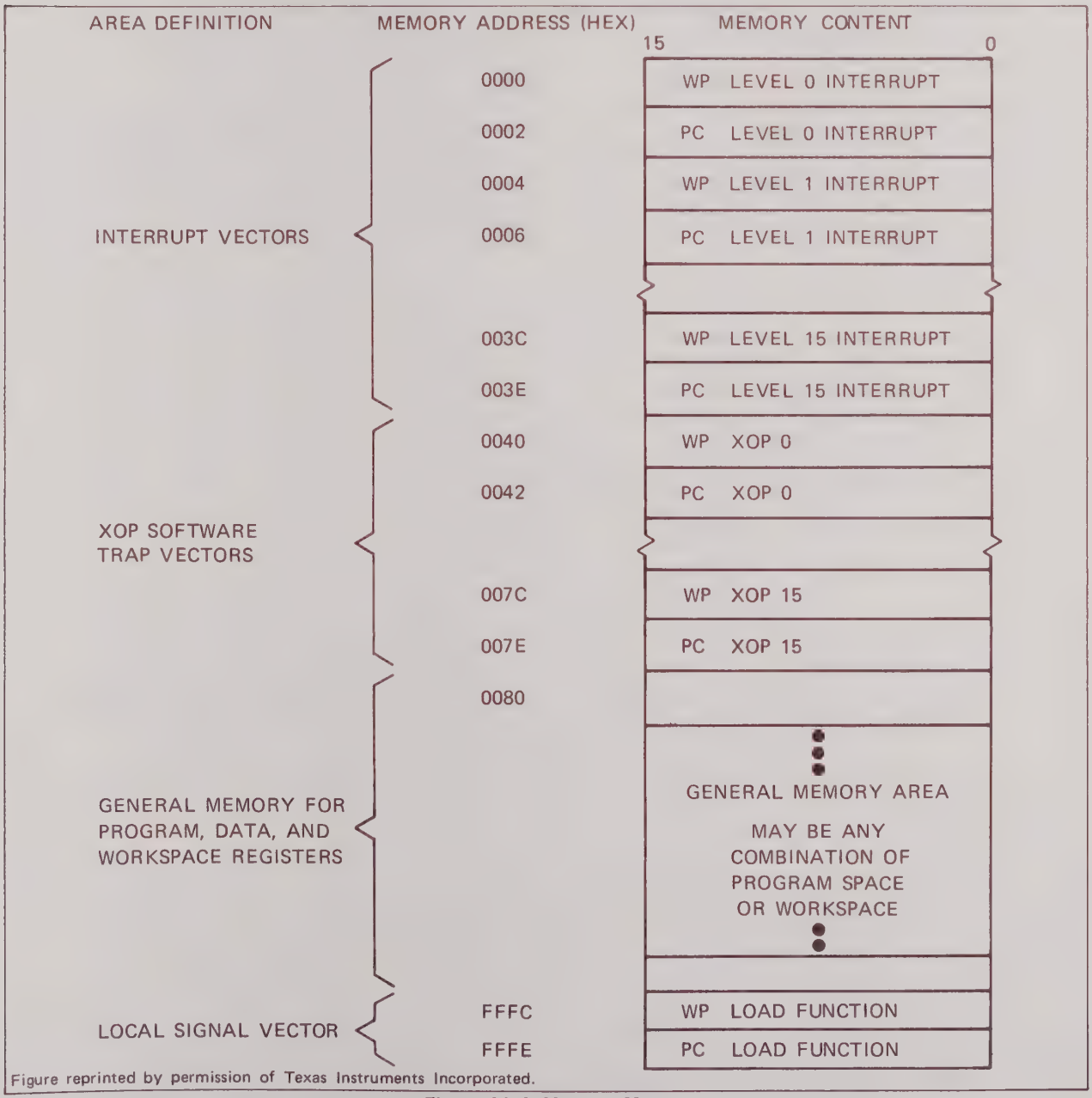


Figure reprinted by permission of Texas Instruments Incorporated.

Figure 2A-3. Memory Map

2A-12. Internal Registers

2A-13. Three internal registers are accessible to the assembly language programmer: the Program Counter, the Status Register, and the Workspace Pointer. The Program Counter (PC) contains the address of the next instruction to be executed. The microprocessor uses this address to fetch the next instruction from memory, and then increments it automatically or sets it to another value if the instruction under execution is a branch. The Status Register (ST) contains status information such as the interrupt mask level and information pertaining to the instruction operation (greater than, equals, overflow, etc.). This register is not to be confused with the status block shown in Figure 2A-2. The Workspace Pointer (WP) contains the address of the first word in the currently active set of workspace registers.

2A-14. Workspace Register File

2A-15. A workspace register file occupies 16 contiguous memory words in the general memory area. The individual workspace registers may hold data or addresses, and function as operand registers, accumulators, address registers, or index registers. During instruction execution, the processor addresses any register in the workspace by adding the register number (times two) to the contents of the workspace pointer and initiating a memory read or write operation. Refer to Figure 2A-4 and 2A-5.

2A-16. This workspace concept is useful during operations that require a context switch, which is a change from one program environment to another (as in the case of an interrupt response or a subroutine call). The microprocessor accomplishes a complete context switch with only three memory write operations and three memory read operations. After the switch, the workspace pointer may contain the starting address of a new 16-word workspace in memory for use in the new routine.

2A-17. Instruction Cycle

2A-18. Figure 2A-6 shows a flow chart of the instruction cycle for the microprocessor. This flow chart presents a pictorial representation of how the microprocessor accomplishes the fetch, decode, and execution of the instructions making up the software. The load and reset operations are described later in this section.

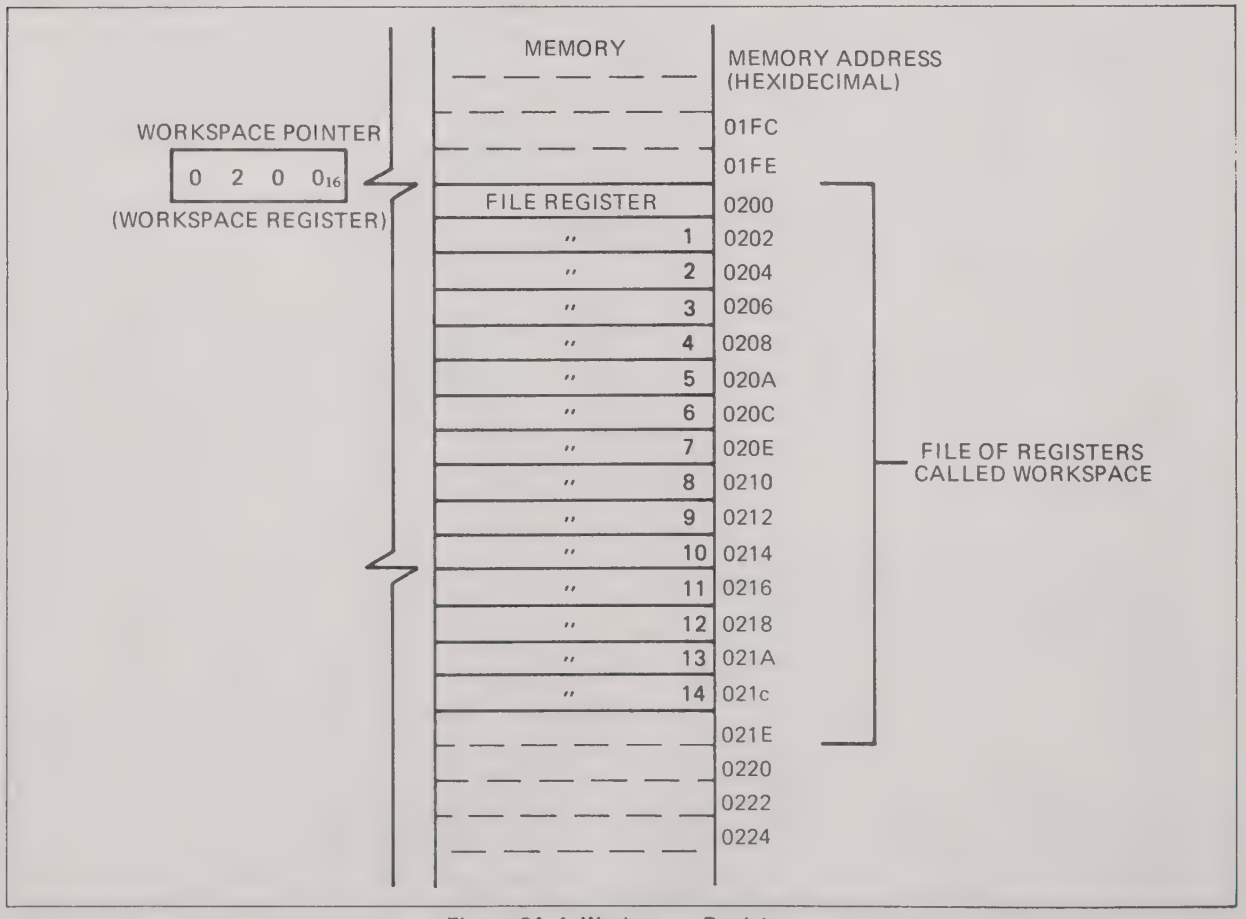


Figure 2A-4. Workspace Registers

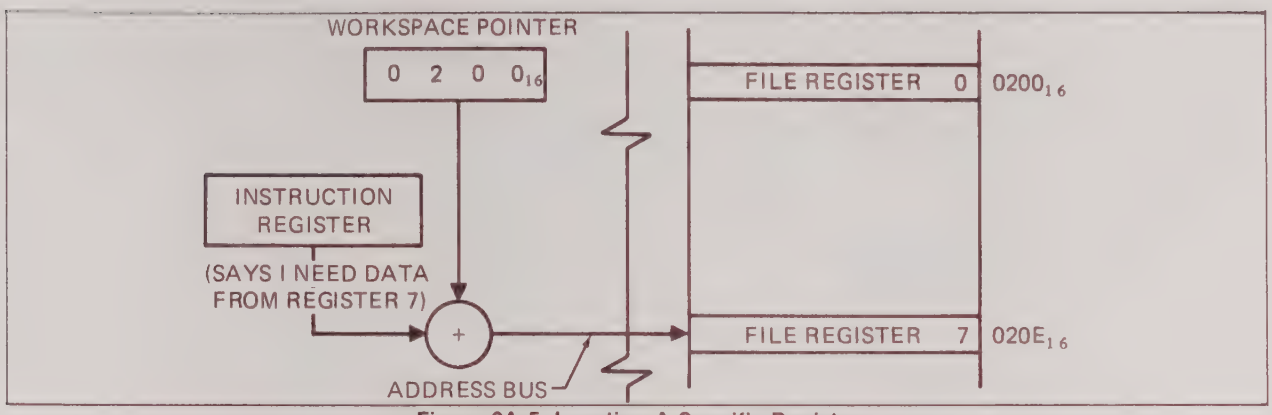


Figure 2A-5. Locating A Specific Register

2A-19. MEMORY DATA TRANSFERS

2A-20. The CPU Module initiates memory data transfers for three different purposes: data read operations, data write operations, and instruction aquisition. Read and write operations are often done in response to instructions in the software that are fetched from memory and decoded in the decoding and execution process. Refer to Figure 2A-7 for descriptions of the various microprocessor signals involved.

2A-21. Memory timing is shown in Figure 2A-8. The CPU Module goes through the following steps during memory data transfers:

1. Memory Enable (MEMEN-) goes active (low) during each memory cycle. At the same time that MEMEN- is active, the memory address appears on the microprocessor outputs AO through A14. Refer to Figure 2A-9. Notice that the microprocessor address bit numbering is reversed from the numbering on the Address Bus. Therefore, Address Bus bits ADRO1 through ADR15 carry the memory address with ADRO1 being the least significant bit and ADR15 being the most significant bit. If the cycle is a memory read cycle, DBIN (Data Bus In) goes active (high) at the same time MEMEN- and AO through A14 become valid. The memory-write signal WE- (Write-Enable) remains inactive (high) during a read cycle. If the read cycle is also an instruction Acquisition cycle, IAQ (Instruction Acquisition) goes high during the cycle. The write cycle discussion is in Step 4.

2. Refer to Figure 2A-2, CPU Module Block Diagram. The memory address goes through the Memory Address Buffer block to the 1720A Address Bus. It also goes to the ROM block and the Scratch Pad RAM. Figure 2A-10, Memory Area Selection, shows how this address is decoded to determine which portion of the addressable memory receives the address. The data from an addressed location within the Memory Module appears on the 1720A Data Bus and then on the Internal Data Bus. Data from a ROM or Scratch Pad RAM location appears directly on the Internal Data Bus. Use the following notes to study Figure 2A-10.

   a. OMS- (On-board Memory Strobe) goes active (low) for either a ROM or Scratch Pad RAM address and disables the Data Buffers to the 1720A Data Bus.

   b. ROM- (Read Only Memory) goes active (low) to enable the on-board ROM.

   c. RAM- (Random Access Memory) goes active (low) to enable the on-board RAM.

   d. IOMEN (Input/Output Memory) goes active (high) for address FXXX and above, to disable the Memory Module by setting address bits AD16 and AD17 true (high). ADOO through AD15 then address on-board memory (ROM and Scratch Pad RAM) or Memory Mapped I/O.

   e. DBIN active (high) gates data from the 1720A Data Bus to the Internal Data Bus through the Data Buffers, when OMS is not active (high).

3. At the end of the read cycle, MEMEN- and DBIN go inactive (high and low, respectively). The address bus may also change at this time; however, the data remains on the Input Bus for one clock cycle after the read cycle.

4. A write cycle is similar to the read cycle with the exception that WE- goes active (low), and valid write data appears on the Data Bus at the same time the address appears. This causes the Scratch Pad RAM to do a write cycle if addressed. DBIN is inactive (low) during a write cycle. Refer to Figure 2A-10. If the address goes off board, the low state of DBIN gates data through the Data Buffers onto the 1720A Data Bus and forces the control signal BUSWR- (Bus Write/Read-) low to indicate a write cycle to external memory devices. Note that WE- does not go out of the CPU Module, and that the Memory Module must generate its own write enable during a write cycle.

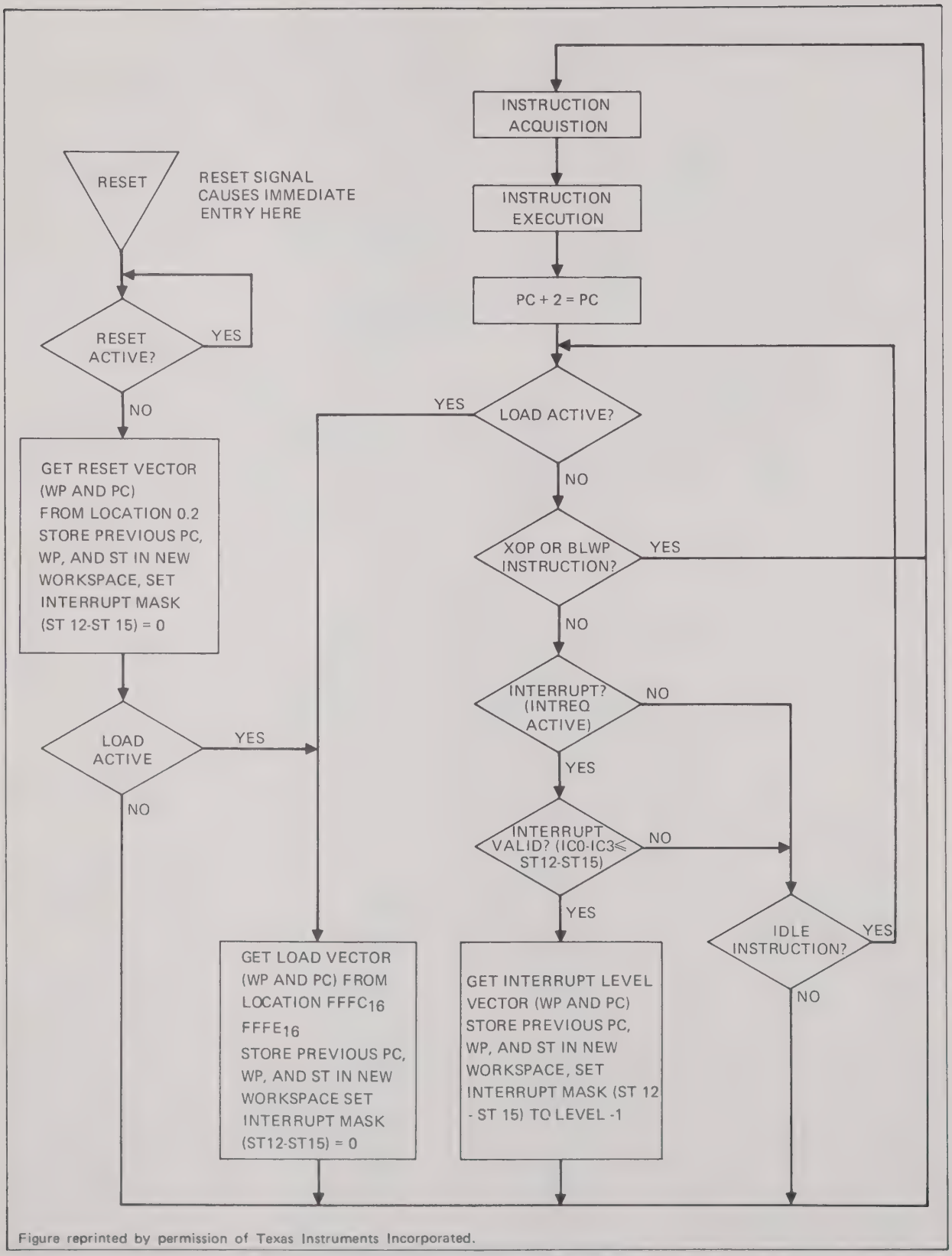

INSTRUCTION ACQUISTION
INSTRUCTION EXECUTION
PC + 2 = PC
RESET
RESET SIGNAL CAUSES IMMEDIATE ENTRY HERE
RESET ACTIVE?
YES
NO
LOAD ACTIVE?
YES
NO
GET RESET VECTOR (WP AND PC) FROM LOCATION 0.2 STORE PREVIOUS PC, WP, AND ST IN NEW WORKSPACE, SET INTERRUPT MASK (ST 12-ST 15) = 0
XOP OR BLWP INSTRUCTION?
YES
NO
INTERRUPT? (INTREQ ACTIVE)
NO
YES
LOAD ACTIVE
YES
NO
INTERRUPT VALID? (IC0-IC3≤ ST12-ST15)
NO
YES
IDLE INSTRUCTION?
YES
NO
GET LOAD VECTOR (WP AND PC) FROM LOCATION FFFC16 FFFE16 STORE PREVIOUS PC, WP, AND ST IN NEW WORKSPACE SET INTERRUPT MASK (ST12-ST15) = 0
GET INTERRUPT LEVEL VECTOR (WP AND PC) STORE PREVIOUS PC, WP, AND ST IN NEW WORKSPACE, SET INTERRUPT MASK (ST 12 - ST 15) TO LEVEL -1

Figure reprinted by permission of Texas Instruments Incorporated.

Figure 2A-6. Microprocessor Flow Chart

SIGNAL	PIN	I/O	DESCRIPTION
			ADDRESS BUS
A0 (MSB)	24	OUT	A0 through A14 comprise the address bus. This 3-state bus provides the memory-address vector to the external-memory system when $\overline{\text{MEMEN}}$ is active and I/O system when $\overline{\text{MEMEN}}$ is inactive. The address bus assumes the high-impedance state when HOLDA is active.
A1	23	OUT	
A2	22	OUT	
A3	21	OUT	
A4	20	OUT	
A5	19	OUT	
A6	18	OUT	
A7	17	OUT	
A8	16	OUT	
A9	15	OUT	
A10	14	OUT	
A11	13	OUT	
A12	12	OUT	
A13	11	OUT	
A14 (LSB)	10	OUT	
			DATA BUS
D0 (MSB)	41	I/O	D0 through D15 comprise the bidirectional 3-state data bus. This bus transfers memory data to (when writing) and from (when reading) the external-memory system when $\overline{\text{MEMEN}}$ is active. The data bus assumes the high-impedance state when HOLDA is active.
D1	42	I/O	
D2	43	I/O	
D3	44	I/O	
D4	45	I/O	
D5	46	I/O	
D6	47	I/O	
D7	48	I/O	
D8	49	I/O	
D9	50	I/O	
D10	51	I/O	
D11	52	I/O	
D12	53	I/O	
D13	54	I/O	
D14	55	I/O	
D15 (LSB)	56	I/O	
			POWER SUPPLIES
VBB	1		Supply voltage (-5 V NOM)
VCC	2,59		Supply voltage (5 V NOM). Pins 2 and 59 must be connected in parallel.
VDD	27		Supply voltage (12 V NOM)
VSS	26,40		Ground reference. Pins 26 and 40 must be connected in parallel.
			CLOCKS
$\phi 1$	8	IN	Phase-1 clock
$\phi 2$	9	IN	Phase-2 clock
$\phi 3$	28	IN	Phase-3 clock
$\phi 4$	25	IN	Phase-4 clock
			BUS CONTROL
DBIN	29	OUT	Data bus in. When active (high), DBIN indicates that the microprocessor has disabled its output buffers to allow the memory to place memory-read data on the data bus during $\overline{\text{MEMEN}}$. DBIN remains low in all other cases except when HOLDA is active.
$\overline{\text{MEMEN}}$	63	OUT	Memory enable. When active (low), $\overline{\text{MEMEN}}$ indicates that the address bus contains a memory address.

Figure 2A-7. Microprocessor Pin Assignments

SIGNAL	PIN	I/O	DESCRIPTION
			BUS CONTROL (cont)
$\overline{WE}$	61	OUT	Write enable. When active (low), $\overline{WE}$ indicates that memory-write data is available from the microprocessor to be written into memory.
CRUCLK	60	OUT	CRU clock. When active (high), CRUCLK indicates that external interface logic should sample the output data on CRUOUT or should decode external instructions on A0 through A2.
CRUIN	31	IN	CRU data in. CRUIN, normally driven by 3-state or open-collector devices, receives input data from external interface logic. When the processor executes a STCR or TB instruction, it samples CRUIN for the level of the CRU input bit specified by the address bus (A3 through A14).
CRUOUT	30	OUT	CRU data out. Serial I/O data appears on the CRUOUT line when an LDCR, SBZ, or SBO instruction is executed. The data on CRUOUT should be sampled be external I/O interface logic when CRUCLK goes active (high).
			INTERRUPT CONTROL
$\overline{INTREQ}$	32	IN	Interrupt request. When active (low), $\overline{INTREQ}$ indicates that an external interrupt is requested. If $\overline{INTREQ}$ is active, the processor loads the data on the interrupt-code-input lines IC0 through IC3 into the internal interrupt-code-storage register. The code is compared to the interrupt mask bits of the status register. If equal of higher priority than the enabled interrupt level (interrupt code equal or less than status register bits 12 through 15) the microprocessor interrupt sequence is initiated. If the comparison fails, the processor ignores the request. $\overline{INTREQ}$ should remain active and the processor will continue to sample IC0 through IC3 until the program enables a sufficiently low priority to accept the request interrupt.
IC0 (MSB) IC1 IC2 IC3 (LSB)	36 35 34 33	IN IN IN IN	Interrupt codes. IC0 is the MSB of the interrupt code, which is sampled when $\overline{INTREQ}$ is active. When IC0 through IC3 are LLLH, the gighest external-priority interrupt is being requested and when HHHH, the lowest-priority interrupt is being requested.
			MEMORY CONTROL
$\overline{HOLD}$	64	IN	Hold. When active (low), $\overline{HOLD}$ indicates to the processor that an external controller (e.g., DMA device) desires to utilize the address and data buses to transfer data to or from memory cycle.* The processor thenplaces the address and data buses in the high-impedance state (along with $\overline{WE}$, $\overline{MEMEN}$, and DBIN) and responds with a hold-acknowledge signal (HOLDA). When $\overline{HOLD}$ is removed, the processor returns to normal operation.
HOLDA	5	OUT	Hold acknowledge. When active (high), HOLDA indicates that the processor is in the hold state and the address and data buses and memory control outputs (WE, MEMEN, and DBIN) are in the high-impedance state.
READY	62	IN	Ready. When active (high), READY indicates that memory will be ready toread or write during the next clock cycle. When not-ready is indicated during a memory operation, the microprocessor enters a wait state and suspends internal operation until the memory systems indicate ready.
WAIT	3	OUT	Wait. When active (high), WAIT indicates that the microprocessor has entered a wait state because of a not-ready condition from memory.

Figure 2A-7. Microprocessor Pin Assignments (cont)

SIGNAL	PIN	I/O	DESCRIPTION
			TIMING AND CONTROL
IAQ	7	OUT	Instruction acquisition. IAQ is active (high) during any memory cycle when the microprocessor is acquiring an instruction. IAQ can be used to detect illegal op codes.
LOAD	4	IN	Load. When active (low), LOAD causes the microprocessor to execute a nonmaskable interrupt with memory address $FFFC_{16}$ containing the trap vector (WP and PC). The load sequence begins after the instruction being executed is completed. LOAD will also terminate an idle state. If LOAD is active during the time RESET is released, then the LOAD trap will occur after the RESET function is completed. LOAD should remain active for one instruction period. IAQ can be used to determine instruction boundaries. This signal can be used to implement cold-start ROM leaders. Additionally, front-panel routines can be implemented using CRU bits as front-panel-interface signals and software-control routines to control the panel operations.
RESET	6	IN	Reset. When active (low), RESET causes the processor to be reset and inhibits WE and CRUCLK. When RESET is released, the microprocessor then initiates a level-zero interrupt sequence that acquires WP and PC from location 0000 and 0002, sets all status register bits to zero, and starts execution. RESET will also terminate an idle state. RESET must be held active for a minimum of three clock cycles.

Figure 2A-7. Microprocessor Pin Assignments (cont)

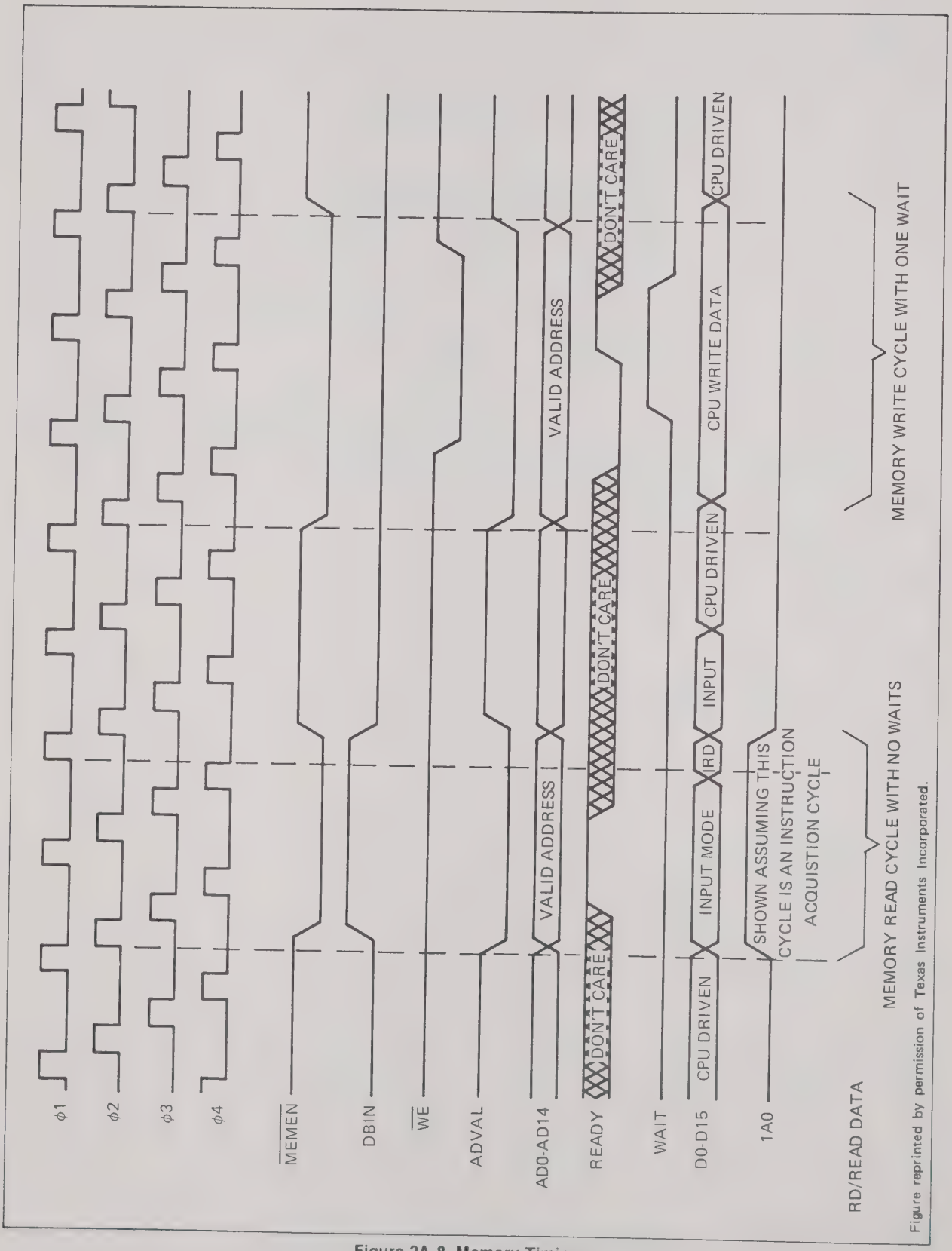


Figure reprinted by permission of Texas Instruments Incorporated.

Figure 2A-8. Memory Timing

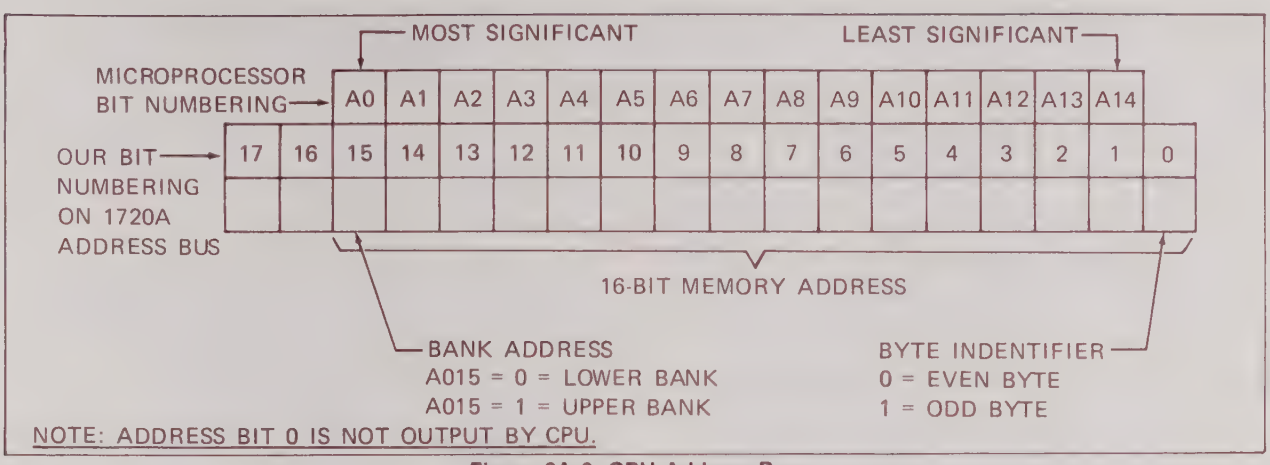

MOST SIGNIFICANT
LEAST SIGNIFICANT
MICROPROCESSOR BIT NUMBERING
A0 A1 A2 A3 A4 A5 A6 A7 A8 A9 A10 A11 A12 A13 A14
OUR BIT NUMBERING ON 1720A ADDRESS BUS
17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
16-BIT MEMORY ADDRESS
BANK ADDRESS
A015 = 0 = LOWER BANK
A015 = 1 = UPPER BANK
BYTE INDENTIFIER
0 = EVEN BYTE
1 = ODD BYTE
NOTE: ADDRESS BIT 0 IS NOT OUTPUT BY CPU.

Figure 2A-9. CPU Address Bus

	ADDRESS IN HEXIDECIMAL		ADDRESS ON ADDRESS BUS																MEMORY SELECTION	ACTIVE SIGNALS
			17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
2K WORDS	3	F F F F	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	CPU ROM	ROM- OMS- IOMEM AD16 · AD17 = 1
	3	F 8 0 0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0		
	3	F 7 F F	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	CPU SCRATCHPAD RAM	RAM- OMS- IOMEM AD16 · AD17 = 1
	3	F 6 0 0	1	1	1	1	1	1	0	1	1	0	0	0	0	0	0	0	0	0		
	3	F 5 F F	1	1	1	1	1	1	0	1	0	1	1	1	1	1	1	1	1	1	MEMORY MAPPED I/O	IOMEM
	3	F 0 0 0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0		AD16 · AD17 = 1
30K WORDS	0	E F F F	0	0	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	MEMORY MODULE UPPER BANK	DBIN- AD15 = 1
	0	8 0 0 0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		AD16 · AD17 = 1
	0	7 F F F	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	MEMORY MODULE LOWER BANK	DBIN- AD15 = 0
	0	0 0 0 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		AD16 · AD17 = 1

Figure 2A-10. Memory Area Selection

2A-22. MEMORY MAPPED I/O OPERATIONS

2A-23. Refer to Figure 2A-10. For the addresses when IOMEM is active and OHM- is inactive (high), the CPU Module still carries out memory operations as described above. The only difference is that the CPU Module sends and receives data to and from I/O devices that respond to these addresses instead of memory. This allows the software to control I/O devices via the 1720A Address and Data Buses. See Electronic Disk Data Transfers for an example of memory mapped I/O operations.

2A-24. CRU IN/OUT DATA TRANSFERS

2A-25. In addition to the Memory Mapped I/O function mentioned previously, the CPU Module can send and receive serial data to and from devices (both on and off the CPU Board) via the CRUIN and CRUOUT lines from the microprocessor.

NOTE

CRUIN becomes CRIN outside the CPU Module, and CRUOUT becomes CROUT outside the CPU Module.

2A-26. The CRU interface timing is shown in Figure 2A-11. In all uses of the CRU (Communications Register Unit), the sequence of events is as shown in the following steps:

1. Address Bits ADR01 through ADR12 carry 1 of 4096 possible addresses. This address goes to the 1720A Address Bus; and with some decoding, to the Status Block; and to the two Serial Ports.

2. If output operation is indicated, CRUCLK, from the microprocessor, goes active (high) and the microprocessor sets the CRUOUT line to a 1 or 0.

NOTE

CRUCLK becomes CRCLK outside the CPU Module.

3. If an input operation is indicated, CRUCLK remains inactive (high) and the addressed device places a 1 or 0 on the CRUIN line.

NOTE

An addressed device external to the CPU Module causes CRACK (CRU Address Acknowledge) to go active (low) when it recognizes the address.

2A-27. The signals CRIN and CROUT can transfer 1 to 16 bits per instruction cycle depending on the software instruction. For example, the two Serial Ports on the CPU Module and the IEEE-488 Module make use of the multi-bit feature and the Clock Module uses the single bit mode.

2A-28. ELECTRONIC DISK DATA TRANSFERS

2A-29. Transfers to and from the Electronic Disk are accomplished as described earlier in Memory Mapped I/0 operations. The concept involved, however, needs some additional explanation.

2A-30. Four memory locations are used to access registers inside the Electronic Disk. Refer to Table 2A-1.

2A-31. To write a file into the Electronic Disk, the software must go through the following steps:

1. Send an I/O Memory Mapped write address to the Electronic Disk Address Register (Register 1), placing the file address on the 1720A Data Bus.

2. Execute one Memory Mapped I/O write operation to the Electronic Disk Data Register (Register 3) for each word in the file. During this step, the electronics in the Electronic Disk automatically increments the Address Register to place each data word into consecutive addresses within the Electronic Disk.

2A-32. To read a data file from the Electronic Disk, the software must go through the following steps:

1. Send a Memory Mapped I/O write address to the Electronic Disk Address Register, placing the file address on the 1720A Data Bus.

2. Execute one Memory Mapped I/O read of the Electronic Disk Data Register for each word in the file.The CPU Module receives the data word on the 1720A Data Bus. While the CPU Module is completing this step, the Electronic Disk automatically increments the Address Register to read consecutive memory locations within the Electronic Disk.

2A-33. Each address (odd and even) in the Electronic Disk accesses one word (two bytes). This is different from the Memory Module where each address accesses one byte. In both cases the 16-bit Address Bus is capable of 64K addresses; however, those addresses contain twice as much data in the Electronic Disk.

2A-34. RESPONSE TO INTERRUPTS

2A-35. This discussion concerns itself with two kinds of interrupts:

1. Device Interrupts during RINT.

2. Reset and Load (non-maskable).

Table 2A-1. Electronic Disk Register Assignments

REG #	HEX ADDRESS	READ/WRITE	FUNCTION
1	F5C0	write	Low order address
2	F5C2	write	High order address
3	F5C6	write/read	Data
4	F5C6	write/read	Status/Control

Note: Address are for board #1. Register #2 is not used on board #1.

2A-36. Device Interrupts During RINT

2A-37. The CPU Module generates the RINT (Refresh-Interrupts) signal at least every 2 ms during the portion of each instruction that is involved with internal data processing (not using the Data or Address Buses). During a hold or a byte mode state, RINT occurs at the clock rate. These states come about when the microprocessor executes certain instructions. RINT goes out onto the 1720A Control Bus. Upon receiving RINT, any device that needs to interrupt the current program being run by the microprocessor pulls the 1720A Data Bus line assigned to it low. This limits the maximum possible interrupting devices to 16.

NOTE

It is only during RINT time that the 1720A Data Bus contains low-true interrupt addresses. Also note that it is during RINT cycles that the Memory Module refreshes the dynamic RAM chips in main memory.

2A-38. In the CPU Module the External Data Bus carrying the address of the interrupting device goes to the Priority Interrupt Logic. See Figure 2A-2. The Priority Interrupt Logic assigns the data bit O the highest priority and gives data bit 15 the lowest priority. The results appear on the four ICx inputs to the microprocessor. If the priority of the interrupting device is higher than the priority of the device the microprocessor is currently processing, the microprocessor does a context switch to the address contained in the corresponding interrupt vector location, and begins executing the service routine for the new device. Eventually, the microprocessor returns to finish processing the first device, providing no new interrupts of greater priority arrive.

2A-39. RESET and LOAD

2A-40. The Reset input to the microprocessor, RDCOK- (Reset DCOK), becomes active upon power on/off because it originates as the 1720A Control Bus signal DCOK (DC OK) from the Power Supply Module. When RDCOK- goes active (low) for a minimum of 3 clock cycles the microprocessor inhibits WE and CRUCLK. When RDCOK- goes high, the microprocessor initiates a level-zero interrupt sequence that acquires WP (Workspace Pointer) and PC (Program Counter) from locations 0000 and 0002. It also sets all status register bits to zero, and starts execution. The Reset input also terminates an idle state.

2A-41. The Load- input to the microprocessor is an output of the Load circuit block and goes active (low) when either of four status bits in the Status circuit are in a set state (Q output = 1). These bits are: HALT, HERR, TOERR, and ACOK (see Status Register later in this section). The active state of Load- occurs at the instruction acquisition time immediately following the setting of the status bit.

2A-42. When Load- goes active (low), the microprocessor executes a nonmaskable interrupt with memory addresses FFFC and FFFE containing the trap vector (WP and PC) after the current instruction is executed. In Addition, the microprocessor will also terminate an idle state. If Load- goes active (low) during the time Reset (RDCOK) is active, the microprocessor executes the LOAD trap after finishing the Reset function. Refer to the flowchart in Figure 2A-6. The Load vector leads to the cold-start loader program stored in read only memory (ROM).

2A-43. RESPONSE TO DMA REQUESTS

2A-44. All DMA devices request access to the 1720A Bus by setting the RQIN (Bus Request In) line to a 1 state (high). Inside the CPU Module, the Hold input to the microprocessor goes active (low) at the next RINT cycle.

2A-45. In response to the Hold signal, the microprocessor places its Address Bus and Internal Data Bus into the high-impedance state (along with WE, MEMEN, and DBIN) and sets HOLDA (Hold Acknowledge) high. This causes the Data Buffers and the Address Buffers in the CPU Module to go into the high impedance state and allow control of the bus by the requesting device (the DMA Floppy)

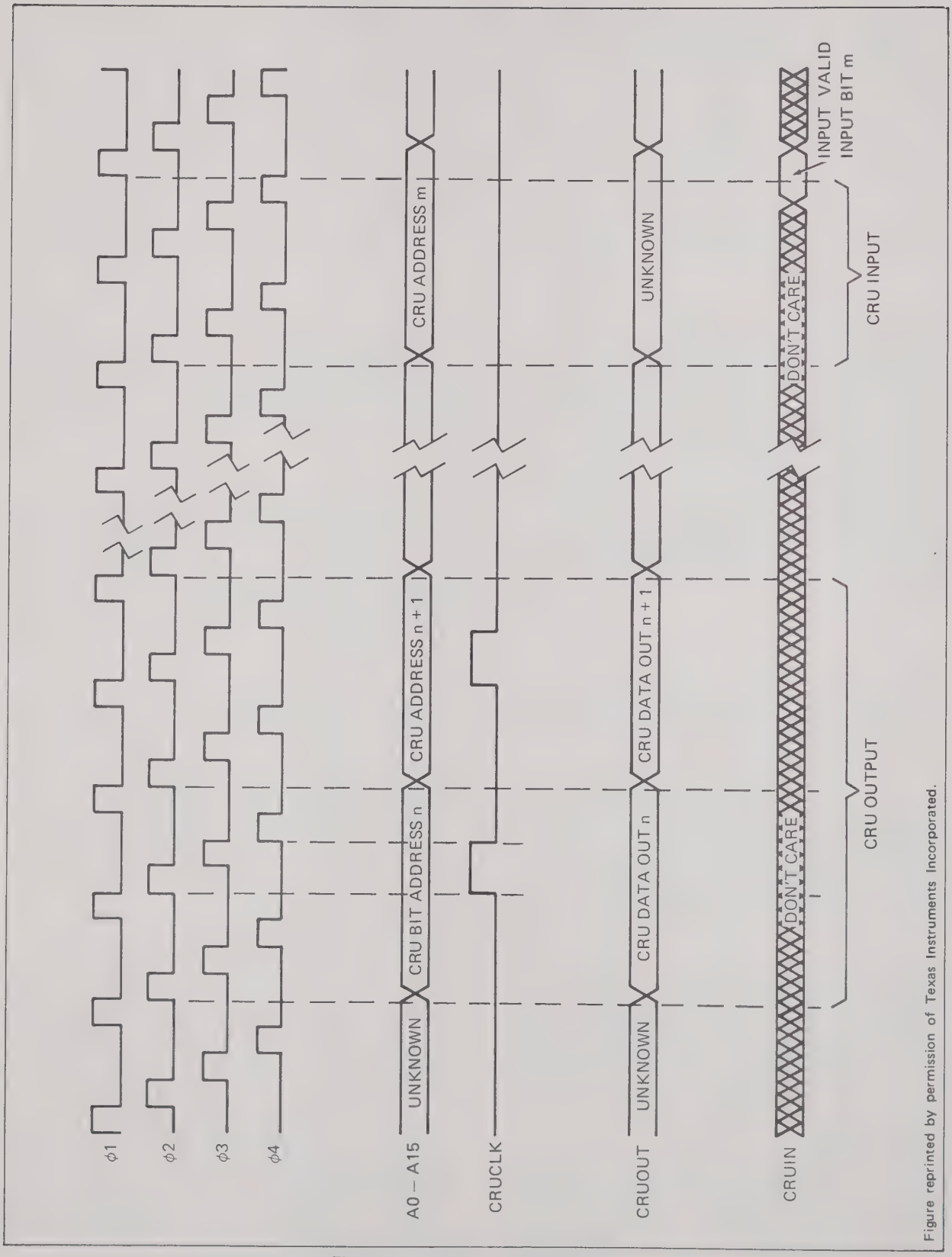


Figure reprinted by permission of Texas Instruments Incorporated.

Figure 2A-11. CRU Interface Timing

2A-46. EXTERNAL INSTRUCTION LOGIC

2A-47. The microprocessor instruction set has five instructions which can be defined by the designer. When the microprocessor executes any of these five instructions, it places a unique three bit code on Address Bus lines AD15, AD14, AD13, and produces a CRICLK pulse. The codes and the 1720A defined functions appear in Table 2A-2.

2A-48. CPU CLOCK AND SYSTEM SYNC

2A-49. The Clock and Timing circuits of the CPU Module produces three groups of outputs. See Table 2A-3.

2A-50. TIME-OUT ERROR

2A-51. The time-out error process insures that the CPU accesses existing memory locations. Any time the CPU addresses memory outside the module and that particular memory device does not respond within a measured time limit, an error condition occurs.

2A-52. Refer to the Time-out Counter block on the schematic. Integreted circuit, U45 is a binary counter that is set to divide by 8. It is driven by 04, but held in the reset state (all zero output) by the high output of U44. This output stays active (high) as long as MEMEN- remains high. As soon as any memory cycle occurs, MEMEN- goes low (address valid) and the counter begins to count. The counter is reset by either OHM- or ADACK- through U44. If neither of these signals occurs before 8 block pulses, the TOERR (Time-out Error) signal goes active (high). This transition sets bit 2 in the Status block.

2A-53. RS-232-C SERIAL PORTS

2A-54. The two RS-232-C Serial Ports are identical and this discussion pertains to both. Refer to the Serial Port block on the schematic. When ADRO8 through ADR12 are all active (high) they enable U36, a decoder, that decodes ADR06 and ADR07, as follows:

ADRO6	ADRO7	OUTPUT ENABLE TO;
X	X	CPU Module Status Register
0	X	Serial Port #1
0	0	Serial Port #2

2A-55. Address bus bits ADR01 through ADRO5 act as inputs to U35 and U34, the Asynchronous Communication Controllers (ACC). The microporcessor uses these lines to select a particular function or mode within and ACC. After this, the ACC is ready to act as a port. When the microprocessor is sending data to the ACC, CRUOUT carries the data and CRUCLK acts as a strobe. When the microprocessor is reading data from theACC, CRUIN carries the data.

2A-56. The 1720A uses the two RS-232-C Serial Ports as data terminals. With this distinction in mind, review the RS-232-C port signal definitions in Table 2A-4.

2A-57. The ACC carries out all the necessary hardware functions for the transmission and reception of data. A software initialization routine uses the address bits ADR01 through ADR05 to set up the ACC internal registers: Control, Interval, Receive Data Rate, and Transmit Data Rate. Following this, the various port signals initiate a mode in the ACC, and the ACC produces an interrupt to the Priority Interrupt Logic. The microprocessor recognizes the interrupt (as discussed previously) and executes one of the software driver routines, shown below, as appropriate:

1. DSR generates an interrupt when going on or off (high or low).

2. CTS enables the transmitter section of the ACC when high.

3. CD same as CTS

4. REC transition from high to low enables the receiver circuitry.

2A-58. CPU MODULE STATUS REGISTER

2A-59. The microprocessor can read and reset any of the four status bits. The chart below describes the bit meanings. Refer to the Status Logic on the CPU Module Schematic and note that the status enable line from U36 in the Serial Port enables both U38 and U39. By executing a CRU operation to the correct address (ADRO6 through ADRO12), the microprocessor can read and reset any bit in the Status Register by placing the appropriate code on ADRO1 through ADRO3. Integrated circuit U38 decodes ADRO1 through ADF03 to select the status bit that goes onto CRUIN. Similarily, U39 decodes ADR01 through ADRO3 to reset that same bit on a later clock pulse.

2A-60. The Status register bits each have a particular meaning. Each is defined in Table 2A-5.

Table 2A-2. External Instructions

ADDRESS BUS			SIGNAL NAME ON 1720A BUS	FUNCTION	uP NAME
AD15	AD14	AD13			
1	1	1	HALT	HALT	LREX
1	1	0	TEST	Bus Test	CKOF
1	0	1	SSTEP	Single Step	CKON
0	1	1	BCLR	Bus Clear	RESET
0	1	0	IDLE	Idle	IDLE
0	0	0	CRU	CRU	CRU

Table 2A-3. Clock And Timing Outputs

GROUP NUMBER	OUTPUT	DESCRIPTION
1	ϕ1 ϕ2 ϕ3 ϕ4	MOS-level (0 and +12V) Four-phase clock output to the microprocessor.
2	1TTL- 2TTL- 3TTL- 4TTI-	TTL-level voltage (0 and 3.5V) Four-phase clock output to the Clock Bus.
3	SYNC	Clock signal for the system.

Table 2A-4. RS-232 Port Signal Definitions

J22/J23 PIN NUMBER	CIRCUIT*	SIGNAL NAME	SIGNAL MEANING	IN/OUT
1	AA	GND	Ground (Cable Shield)	--
2	BA	XMT	Transmit Data	OUT
3	BB	REC	Receive Data	IN
4	CA	RTS	Request To Send	OUT
5	CB	CTS	Clear To Send	IN
6	CC	DSR	Data Set Ready	IN
7	AB	RET	Signal Return (common)	--
8	CF	CD	Carrier Detect	IN
9	CD	DTR	Data Terminal Ready	OUT

* Taken from EIA Standard RS-232-C

Table 2A-5. CPU Status Register

BIT #	CONTROL SIGNAL	MEANING
4	DCOK	DC Supplies ready. Goes high approximately 0.5 seconds after power up, and goes low with any power shut down command.
3	ACOK	AC ready. Goes low as line power drops out for more than 30 ms or the ABORT switch in the Power Supply Remote Connector is set.
2	TOERR	Time-out Error.
1	HERR	Hardware Error. Usually the result of a detected memory parity error.
0	HALT	Halt

Note: The setting of S1, Baud Rate, is an input to U39 and is available during a read of the Status Register.

Section 2B
Memory Module

2B-1. INTRODUCTION

2B-2. The Memory Module provides a storage medium, for both programs and data, from which programs can be executed. A block diagram of the Memory Module is shown in Figure 2B-1.

2B-3. Certain words used in this description require definition. These words, and their meanings are:

1. A bit is a single binary digit.
2. A byte is 8 bits.
3. A word is 2 bytes.
4. K is the number 1024.

2B-4. MAIN MEMORY

2B-5. Dynamic RAM

2B-6. The 36 dynamic RAM ICs in the Main Memory each have 16K addressable locations that can hold 1 bit of information. Because these chips are dynamic, the Memory Module must perform a refresh operation every 2 ms to keep this information valid. The Transparent Refresh portion of this discussion covers the refresh operation in detail.

2B-7. Parallel Address And Data Lines

2B-8. Parallel address and data lines connect the memory ICs into one large array. This means that all ICs receive the same address and data inputs. A system of strobes determine exactly which ICs accept a given address and data. The on board strobe generation and the byte/word addressing portions of this discussion cover the use and generation of these strobes.

2B-9. Independent Upper And Lower Address Banks

2B-10. The independent upper and lower address banks each have a 16K word capacity. Refer to the main memory block diagram in Figure 2B-2 and note that it takes 1 RASxx strobe to access each byte in a given bank. During word mode operations, both of these strobes occur at the same time, and during byte mode they occur separately. The Byte/Word Addressing portion of this discussion covers this process in detail. Refer to Table 2B-1 for a definition of the strobe mnemonics. Then refer to Figure 2B-3 for an annotated layout of the RAM ICs that comprise the Main Memory.

2B-11. 7-Bit Chip Address Bus

2B-12. The 7-bit Chip Address Bus, which is the output of the Row/Column Decoder, provides the row/column address input to the Main Memory. Under the direction of the Row/Column Decoder, this bus first carries 1720A Address Bus bits AD08 through AD14 to make up the column address. It then carries bits AD01 through AD07 to make up the row address. As discussed earlier, the column address strobes and the row address strobes determine the chips which accept these addresses.

2B-13. Read And Write Operations

2B-14. The read and write operations both begin as soon as the chips receive the row address strobe. At this time, internal decoding within either 9 chips (byte mode) or 18 chips (word mode) in the addressed bank access the correct 1 out of 16K addresses within that bank. See the discussion of Byte Parity for an explanation of the extra bits in a byte and a word.

2B-15. During a read operation the Main Memory places the contents of the data locations onto the Data Output Bus and sends the contents of the parity locations to the Parity Generator/Checker.

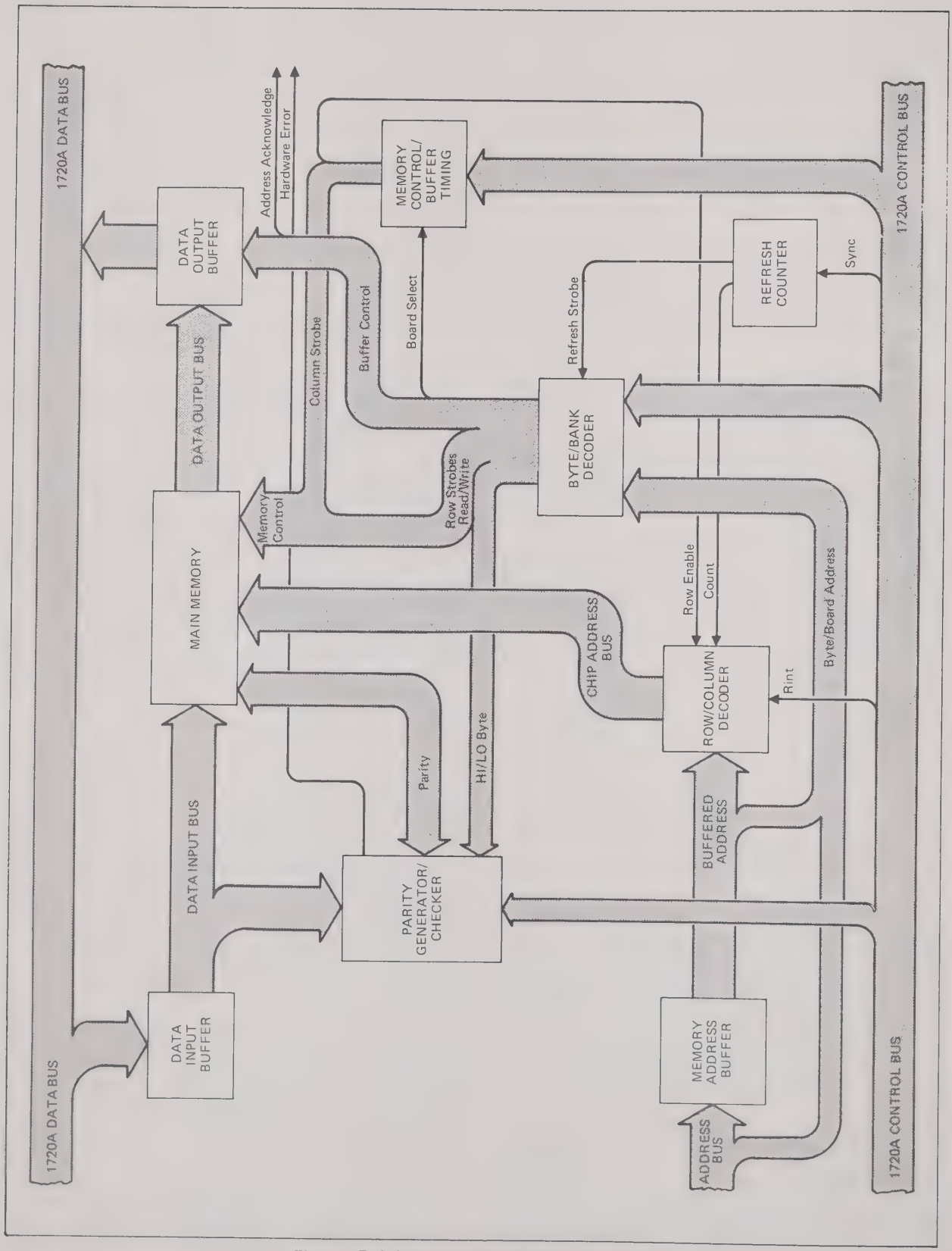


Figure 2B-1. Memory Module Block Diagram

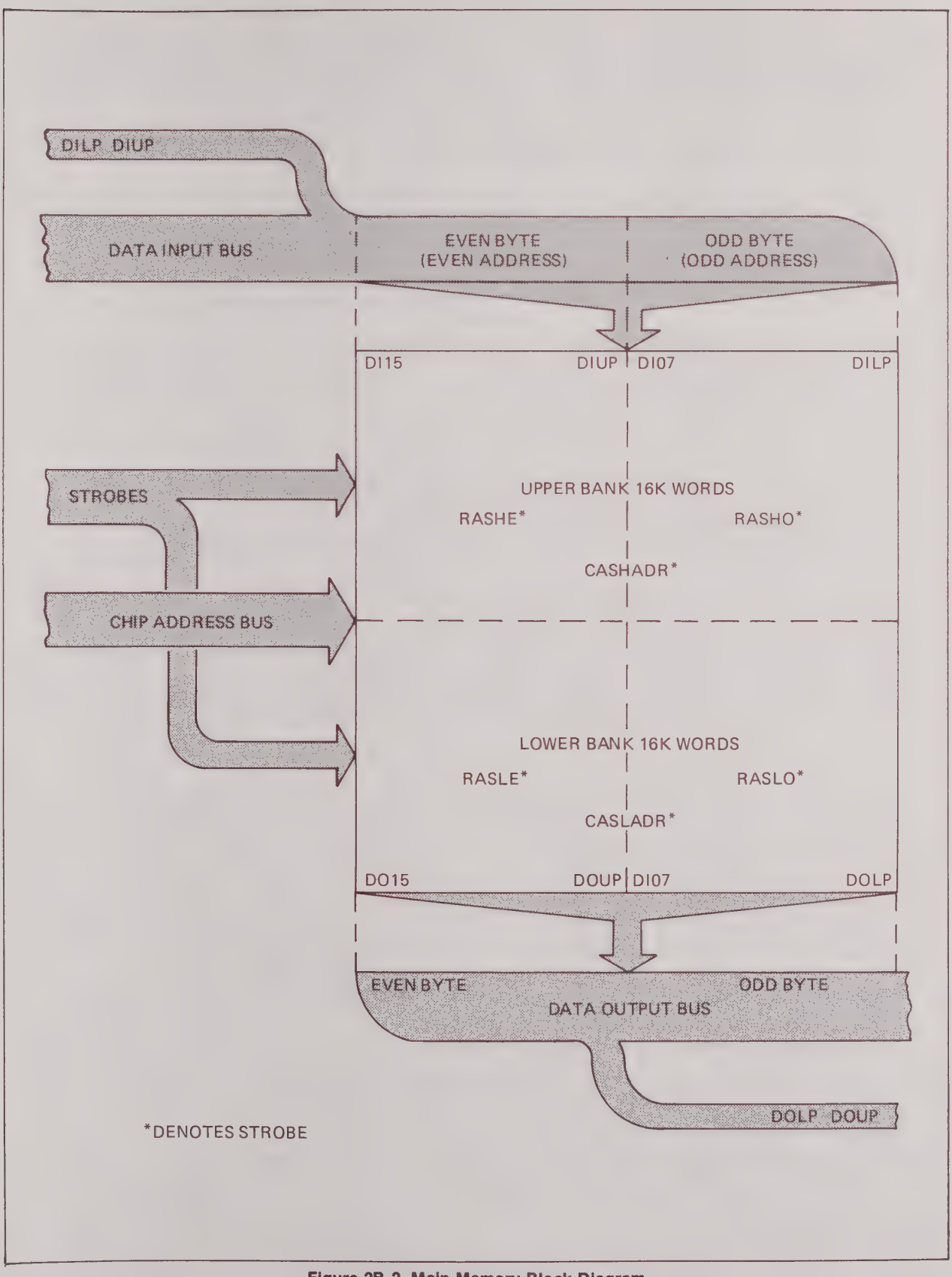


Figure 2B-2. Main Memory Block Diagram

Table 2B-1. Strobe Mnemonics

STROBE	MEANING
CASLADR-	Column Address Strobe, Low Addresses.
CASHADR-	Column Address Strobe, High Addresses.
RASLO-	Row Address Strobe, Low address bank, Odd byte.
RASLE-	Row Address Strobe, Low address bank, Even byte.
RASHO-	Row Address Strobe, High address bank, Odd byte.
RASHE-	Row Address Strobe, High address bank, Even byte.

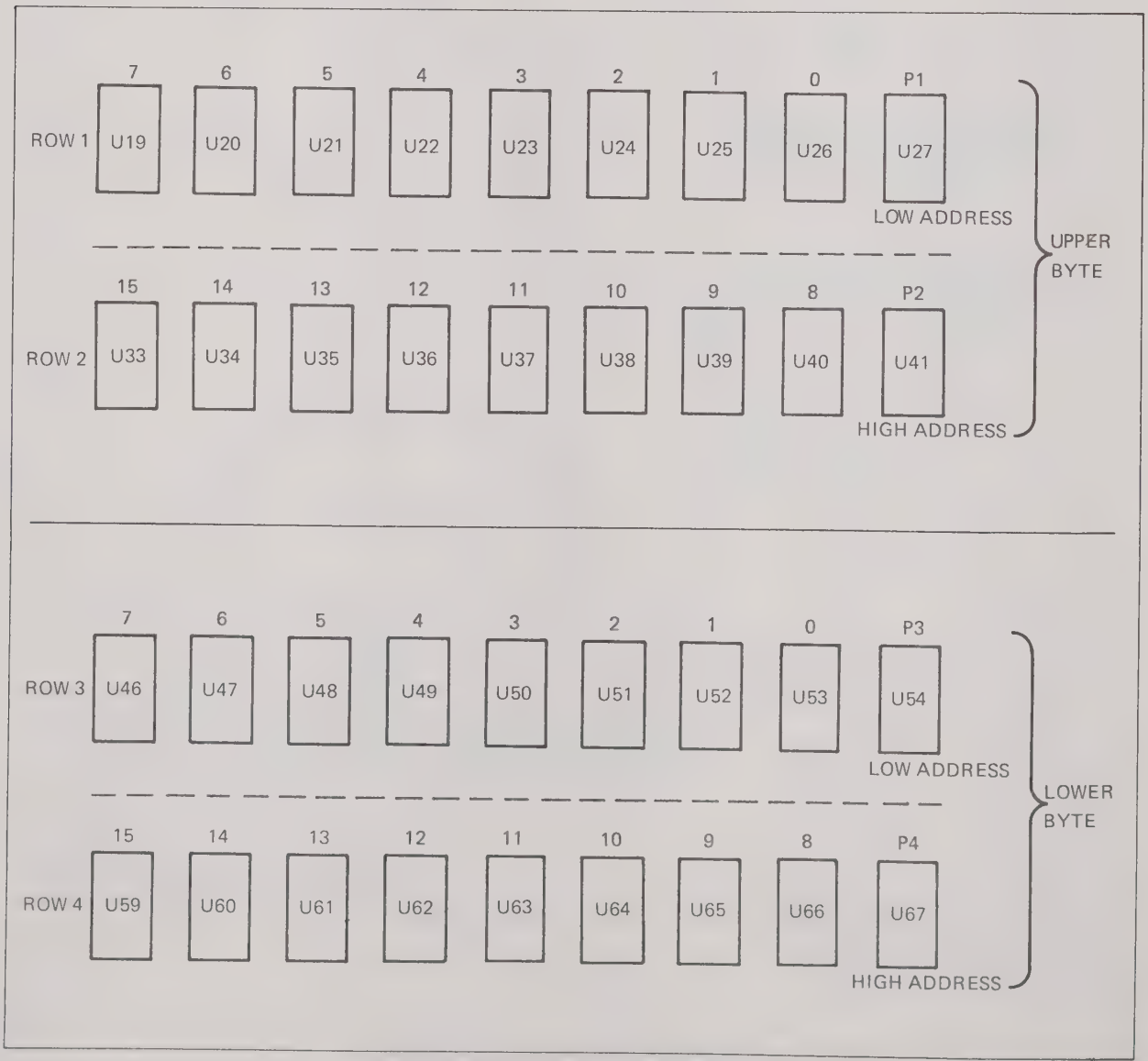


Figure 2B-3. Main Memory Chip Layout

2B-16. During a write operation, the Main Memory replaces the contents of the data locations with the contents of the Data Input Bus. It also replaces the contents of the parity locations with the output of the Parity Generator/Checker.

2B-17. BYTE/WORD ADDRESSING

2B-18. The Memory Module uses the register address strobes and the data out signals to accomplish byte/word addressing. There are 4 lines coming into the Memory Module that initiate and control this activity. They are:

1. BMODE- Byte MODE control (0 = byte mode)
2. BUSWR- Bus Write/Read (0 = write)
3. AD00 Address bit 0 (1 = odd byte)
4. AD15 Address bit 15 (1 = upper bank)

2B-18a. Refer to Table 2B-2 for the various strobes and output control signals that accomplish the actual byte/word addressing and control functions. See the Data Output Buffer discussion for an explanation of the mnemonics for DOE-, D00B-, and D00W-. Refer to the Byte/Bank Decoder section of the Memory Module Schematic for the logic elements involved.

TABLE 2B-2. Byte/Word Addressing and Control

BMODE	RASxx-/DOxx	AD00	AD15	BUSWR
=0 (byte)	RASLO	1	0	=0 (write)
	RASLE	0	0	
	RASHO	1	1	
	RASHE	0	1	
=1 (word)	RASLO-/RASLE	x	0	
	RASHO-/RASHE	x	1	
=0 (byte)	DOE	0	x	=1 (read)
	DOOB	1	x	
=1 (word)	DOE	x	x	
	DOOW-	x	x	
		x = don't care		

2B-19. TRI-STATE BUFFERS

2B-20. There are three sets of tri-state buffers in the Memory Module. Each set is described in the paragraphs that follow.

2B-21. Memory Address Buffer

2B-22. The Memory Address Buffer output goes to the Row/Column Decoder (AD01 - AD14), and to the Byte/Bank Decoder (AD00 and AD15). AD00 identifies bytes and AD15 identifies address banks.

NOTE

Although the buffers in the Address Buffer are the tri-state type, they function only as conventional buffers and are not used as tri-state devices.

2B-23. Data Input Buffer

2B-24. The Data Input Buffer operates in the word mode and in the byte mode. Turn to the Schematic (Data Input Buffer) and note the influence of the byte-mode control signal (BMODE-), as shown in Table 2B-3. The word-mode operation requires no explanation.

Table 2B-3. Data Input Buffer Mode Control

MODE	BMODE- STATE	BUFFERS USED
Word	1	U18, U42
Byte	0	U18, U45

2B-25. During byte mode, only the upper 8 bits of the 1720A Data Bus have data. The ground on pins 1 and 19 of U18 provides a constant enable. Therefore the data byte on the upper 8 bits of the 1720A Data Bus appears on both the lower and the upper 8 bits of the Data Input Bus (inside the Memory Module). This provides the same data input to both the odd and even sets of memory chips at the addressed location. Refer to Table 2B-2 and notice that BMODE-, AD00, and AD15 control which set of chips receive a RASxx strobe and store the data.

2B-26. Data Output Buffer

2B-27. The Data Output Buffer operates in word or byte mode as determined by the buffer control lines. During byte mode, only the upper 8 bits of the 1720A Output Bus carry data. This data can come from either the even or odd byte of a word location in the Main Memory. Each of the 3 buffers that make up the Data Output Buffer has a separate enable line. Refer to Table 2B-4, for an explanation of the Data Output Buffer Control lines.

Table 2B-4. Data Output Buffer Control

SIGNAL	FUNCTION
DOE-	Data output is even. Transfers a byte to 1720A Data Bus bits DI08--DI15.
DOOB-	Data output is odd, byte mode. Transfers a byte to 1720A Data Bus bits DI08--DI15.
DOOW-	Data output is odd, word mode. Transfers a byte to 1720A Data Bus bits DI00--DI07 in word mode.

2B-28. ON BOARD STROBE GENERATION

2B-29. The two circuits responsible for strobe generation are the Byte/Bank Decoder and the Memory Control/Buffer Timing. The discussion that follows covers these blocks.

2B-30. Byte/Bank Decoder

2B-31. The Byte/Bank Decoder has outputs that fall into six functional groups:

1. Address acknowledge signals
2. Word/byte mode control
3. Row address strobes
4. Parity check strobes
5. Output buffer control
6. Memory read/write control

2B-32. The address acknowledge signals ADACK- (Address Acknowledge) and BRDSEL (Board Select) act as follows:

1. ADACK- goes out to the 1720A Control Bus and is active when a given address is in the range of the Memory Module (AD16 and AD17 = 0).

2. BRDSEL occurs at the same time as ADACK- and initiates a memory operation in the Memory Control Buffer and Timing.

2B-33. The byte mode control output signal BMODE- (Byte Mode) controls buffer usage in the Data Input Buffer. Note that as an input signal to the Byte/Bank Decoder, BMODE- sets up the row address strobe outputs for the byte or word mode. See Table 2B-2.

2B-34. The Row address strobe signals (RASLE-, RASHO, and RASLO-) are described in Table 2B-1 and Table 2B-2.

2B-35. The Parity check strobes LOADDER- (Low Address Read) and HIADDER- (High Address Read) act as inputs to the Parity Generator/Checker, and strobe the two parity bits into the check circuits on a read operation.

2B-36. The Output buffer control signals (DOE-, D00B-, and D00W-) act as inputs to the Data Output Buffer, and control buffer usage as described earlier in this section.

2B-37. The Memory read/write control signals R/WHADR- (Read/Write High Address) and R/WLADR- (Read/Write Low Address) cause a read or write operation in the Main Memory block. They are set to a 0 state for a read and to a 1 state for a write.

2B-38. Memory Control Buffer And Timing

2B-39. As previously explained earlier under Independent Upper and Lower Address Banks, a sequence of events takes place during each memory operation. This sequence is under the direction of the Memory Control Buffer and Timing, which has four outputs. These outputs are:

1. RASEN is the Row Address Strobe Enable to the Byte/Bank Decoder.

2. RWEN (Row Enable) controls the Row/Column Decoder output to the Chip Address Bus. It is a 1 for the row address and a 0 for the column address. Refer to the Chip Address Multiplexing discussion later in this section.

3. CASLADR- is the Column Address Strobe Low Address to the Main Memory

4. CASHADER- is the Column Address Strobe High Address to the Main Memory.

2B-40. Refer to the Memory Control Buffer and Timing on the Memory Module schematic, and note that both column address strobes occur at the same time since they are outputs from two inverters (U12) which receive the column address each time the Main Memory is accessed (except during refresh operations).

2B-41. CHIP ADDRESS MULTIPLEXING

2B-42. The multiplexing feature of the memory chips allows the use of a 14-bit address, but requires only seven pins on the ICs. Chip address multiplexing of the incoming address occurs within the Row Column Decoder.

2B-43. Refer to the Row Column Decoder on the Memory Module Schematic. Multiplexer U57 responds to the state of the RWENA (Row Enable) line from the Memory Control Buffer and Timing block as follows:

1. RWEN = 0 Places the upper bits (AD08 - AD14) onto the Chip Address Bus (column address).

2. RWEN = 1 Places the lower bits (AD01 - AD07) onto the Chip Address Bus (row address).

2B-44. TRANSPARENT REFRESH

2B-45. To keep the Main Memory contents valid, a refresh of the dynamic cell matrix within each memory chip must be accomplished in each 2 ms time interval. This is done by performing a memory cycle at each of the 128 row addresses inside each chip. Although any normal memory cycle performs this refresh operation, row-address-only cycles most easily accomplish the function. Because the refreshing takes place during microprocessor RINT (Refresh/Interrupt) cycles when the CPU is not using the memory, the entire operation is transparent.

2B-46. Two circuits on the Memory Module are responsible for the refresh operation, the Refresh Counter and the Row Column Decoder. Both are discussed in the following paragraphs.

2B-47. Refresh Counter

2B-48. The Refresh Counter has two outputs which control the refreshing of the dynamic RAM ICs in the Main Memory block during microprocessor RINT cycles:

1. COUNT goes to a 0 state to increment the row refresh counter in the Row/Column Decoder.

2. REFSTRB-(Refresh Strobe) goes to the Byte/Bank Decoder and causes all four row address strobes to occur at once.

2B-49. Row Column Decoder

2B-50. The Row Column Decoder handles the refreshing of the dynamic RAMs in the Main Memory in addition to the multiplexing operation described earlier. Refer to the schematic and note that the signals RINT and COUNT are inputs to the multiplexer U57 in the Row/Column Decoder. These signals perform the following functions:

1. COUNT increments an internal counter which determines the row address to be cycled.

2. RINT, generated by the microprocessor, enables refreshing of one row address.

2B-51. The refresh controller uses these signals within the Row/Column Decoder circuits to cycle through all 128 row addresses every 2 ms. Because the Byte/Bank Decoder generates all the row address strobes simultaneously, each time the COUNT signal is active, both banks in the Main Memory are refreshed at the same time. Note that the RAM IC's receive only the row address strobes. Therefore, they only perform a refresh operation and not a read or a write operation.

2B-52. BYTE PARITY

2B-53. The Parity Generator/Checker performs two functions: Write Parity Generation, Read Parity Checking. These two functions are part of an error check/detection scheme that uses an extra (ninth) bit for each byte. Both functions are described in detail in the following paragraphs.

2B-54. Write Parity Generation

2B-55. To generate parity bits for each byte on a write operation, the Parity Generator/Checker looks at the 16-bit word on the Data Input Bus and treats it as two 8-bit bytes. At this point in time, each byte is treated as if it had 9 bits and the ninth bit (the parity bit) is a 0. It analyzes the lower (even) byte and causes a parity output bit to be set to either a one or zero, as necessary, to maintain an odd number of 1 bits in the byte. At the same time, it analyzes the upper (odd) byte and sets the other parity output to maintain an odd number of 1 bits in that byte. These two parity output bits, DNEP and DNOP (Data In Even Parity and Data In Odd Parity) make up an input to the Main Memory which treats them as part of the data word and stores them along with each byte.

2B-56. Read Parity Checking

2B-57. To check the parity of each byte during a read operation, the Parity Generator/Checker checks the number of 1 bits in each byte (including the parity bit) of the word and produces a Hardware Error signal (HERR) if the correct parity is not found. Note that the two parity bits DOEP and DOOP (Data Out Even Parity and Data Out Odd Parity) are not part of the Data Output Bus and, therefore, do not leave the Memory Module.

2B-58. SWITCH SELECTABLE BANKS

2B-59. The four switches (S1 S2 S3 S4) on the Byte/Bank Decoder establish the board number. However, the 1720A uses only one memory board. S1 and S2 are always on.

Section 2C
IEEE-488 Interface Module

2C-1. INTRODUCTION

2C-2. This section consists of two parts. The first part is an introduction to the IEEE-488 Bus. This part may be skipped if an introduction is not needed. The second part is the actual theory of operation for the IEEE-488 Interface Module. This module provides the two independent IEEE-488 ports that allow the 1720A to be an instrument controller. There can be 15 instruments, including the 1720A, on each port.

2C-3. INTRODUCTION TO THE IEEE-488 BUS

2C-4. The various terms and concepts used in the IEEE Standard 488-1978 entitled "IEEE Standard Digital Interface for Programmable Instrumentation" are described in the following order:

1. Basic Concepts and Terms
2. Bus Operating Modes
3. Three Wire Handshake
4. Operational Sequences

2C-5. Basic Concepts and Terms

2C-6. The IEEE-488 Bus consists of 16 signal lines. All signals on these lines are TTL level, low-true digital logic. Specifically:

1. True is a low level, 0 to 0.8V and is a logic 1.
2. False is a high level, 2 to 5V and is a logic 0.

2C-7. This discussion uses the above definitions in a particular convention. References within text use the terms true and false. Diagrams with waveforms indicate low and high levels. Charts or tables use 0 and 1 or low and high.

2C-8. Bus devices may have a variety of internal functions, and several types of communication capability. However, each must have at least a talker, a listener, or a controller interface capability. Each capability is defined as follows:

1. A Talker is a source of data going to other devices. For example, a voltmeter is talking when it sends measurement results orstatus information to the controller. Only one talker may be active at a time. There are two ways a device may talk. Either an operator switches a device to talk only which causes it to talk all the time, or the controller commands the device to talk.

2. A Listener is an acceptor of data from other devices. For example, a printer is a listener when it receives measurement results from a voltmeter to be printed. Up to 14 devices may listen simultaneously. There are two ways a device may become a listener. Either an operator switches the device to listen-only which causes it to listen all the time, or the Controller commands the device to listen.

3. A Controller manages bus communications. It has the power to designate the talker and listeners on the bus. The 1720A is a System Controller and has the capabilities listed below:

   a. Command devices to listen, talk, or perform their functions.

   b. Trigger devices to perform a pre-programmed function.

   c. Clear devices to an initial state (defined by the device).

d. Poll devices for their status serially (one device at a time).

e. Poll devices for their status in parallel (all devices at once).

f. Command devices to abort current operations.

g. Command devices to enter the remote mode of operation.

2C-9. The devices that are talkers and listeners on the IEEE-488 Bus usually have address switches. The operator sets these switches to the lower 5 bits of the preferred listen/talk address of the device. Refer to the Command Mode discussion and Table 2C-4 under Address Commands for a complete explanation of how the Controller makes use of this address.

2C-10. In addition to having talk, listen, and/or control capabilities, the bus devices can have various additional interface function capabilities designed into them. Table 2C-1 lists all IEEE Interface functions and their symbols. Note that the Controller abilities listed earlier need matching abilities in the bus devices if the Controller ability is to be implemented for a given device.

Table 2C-1. IEEE Interface Function Repertoire

INTERFACE FUNCTION	SYMBOL
Source Handshake	SH
Acceptor Handshake	AH
Talker or Extended Talker	T or TE
Listener or Extended Listener	L or LE
Service Request	SR
Remote Local	RL
Parallel Poll	PP
Device Clear	DC
Device Trigger	DT
Controller	C

2C-11. The 16 IEEE Bus lines fall into three categories. The signals within each category are defined in Table 2C-2. These categories are:

1. Eight data lines.

2. Three handshake lines

3. Five bus management lines.

2C-12. Figure 2C-1 presents a block diagram of bus device connections. The 1720A is connected to the IEEE Bus in the same way as the devices it controls. This reduces the number of device positions from 15 to 14.

2C-13. Bus Operating Modes

2C-14. INTRODUCTION

2C-15. The bus operates in either the command mode or the data mode during the transfer of data from one device to another. A controller uses the command mode to control the various devices on the bus with interface dependent commands. A talker or a controller uses the data mode to transfer data or device dependent commands on the DIOx lines. The three handshake signals mentioned previously officiate transfers on the bus. The handshake portion of this discussion covers this in detail later on.

2C-16. COMMAND MODE

2C-17. A controller places the system into Command Mode by bringing the ATN- line true (low). While ATN- is true, all devices on the bus must interpret the data byte as a command message. Only a controller may issue commands, and the IEEE-488 Standard defines all available command messages. Table 2C-3 defines four categories of command messages. Table 2C-4 defines the command messages within these categories. The codes for the various addresses and command groups fall into numerical groupings. The actual codes and addresses in ASCII, hexadecimal, octal, decimal, and binary appear in the Programming Manual.

2C-18. DATA MODE

2C-19. The controller places the system into the data mode by setting the ATN- line false (high). While ATN- is false, all devices on the bus treat the data byte as data. This data can originate from either the talker or the controller. The data that flows from device to device on the bus (talker to listener) can be in any mutually understood code or format. The 1720A can act as an interpreter, accepting data from a talker and sending it out to a listener. This is covered in detail under Operational Sequences.

2C-20. Three-Wire Handshake

2C-21. All source and accepter bus devices use a three-wire handshake to exchange data bytes. Typically, the source is a talker and the acceptor is a listener. Figure 2C-2 is a flowchart of the handshake process which actually shows two flows, one for the source role and one for the acceptor role. The dotted lines crossing between these two flows show both the direction of the signal and the meaning of the level (high or low) the signal has at that particular time. The circled numbers inside some of the flowchart blocks refer to the level changes shown in the handshake signal line timing sequence, Figure 2C-3.

Table 2C-2. IEEE Bus Signal Lines

CATEGORY	NAME	DEFINITION	FUNCTION
1	DIO 1- DIO 8-	Data Input/ Output 1 through 8	Data, address, or command, in bit-parallel, byte-serial format. Information transfer is asynchronous and bidirectional.
2	DAV-	Data Valid	Low from source to indicate valid data on the DIOx lines. Source won't change data until NDAC- goes high. High from source to acknowledge that NDAC- went high and data is no longer valid.
	NRFD-	Not ready For Data	Wired OR output from all acceptors. Low indicates that one or more acceptors acknowledge that valid data is on the DIOx lines and that they are proceeding to read it. High indicates that all acceptors are ready for more data.
	NDAC-	Not Data accepted	Wired OR output from all acceptors. Refer to Figure 2C-2 (Handshake Flowchart) for full explanation of low state. Low, during the handshake, indicates one or more acceptors are in the process of reading the data and the source must not change it. High indicates that all acceptors have accepted the data and the source may change the data. Low, at the end of the handshake, indicates that all acceptors have acknowledged DAV- going high.
3	ATN	Attention	Low set by Controller to indicate DIOx lines carry addresses or commands from the Controller. High set by Controller to indicate that DIOx lines carry data from talker or Controller.
	IFC	Interface Clear	Pulsed low by Controller to place talkers and listeners into device determined idle state (neither talker nor listener).
	SRQ-	Service Request	Low sent by any capable device to the Controller to indicate a need for service.
	REN	Remote Enable	Low set by Controller to a listener to disable local control and allow the Controller to program the device.
	EOI	End or Identify	Low from talker to indicate end of mutiple byte data transfer (ATN- set high). Low from the Controller to indicate a parallel poll sequence (ATN- set low).

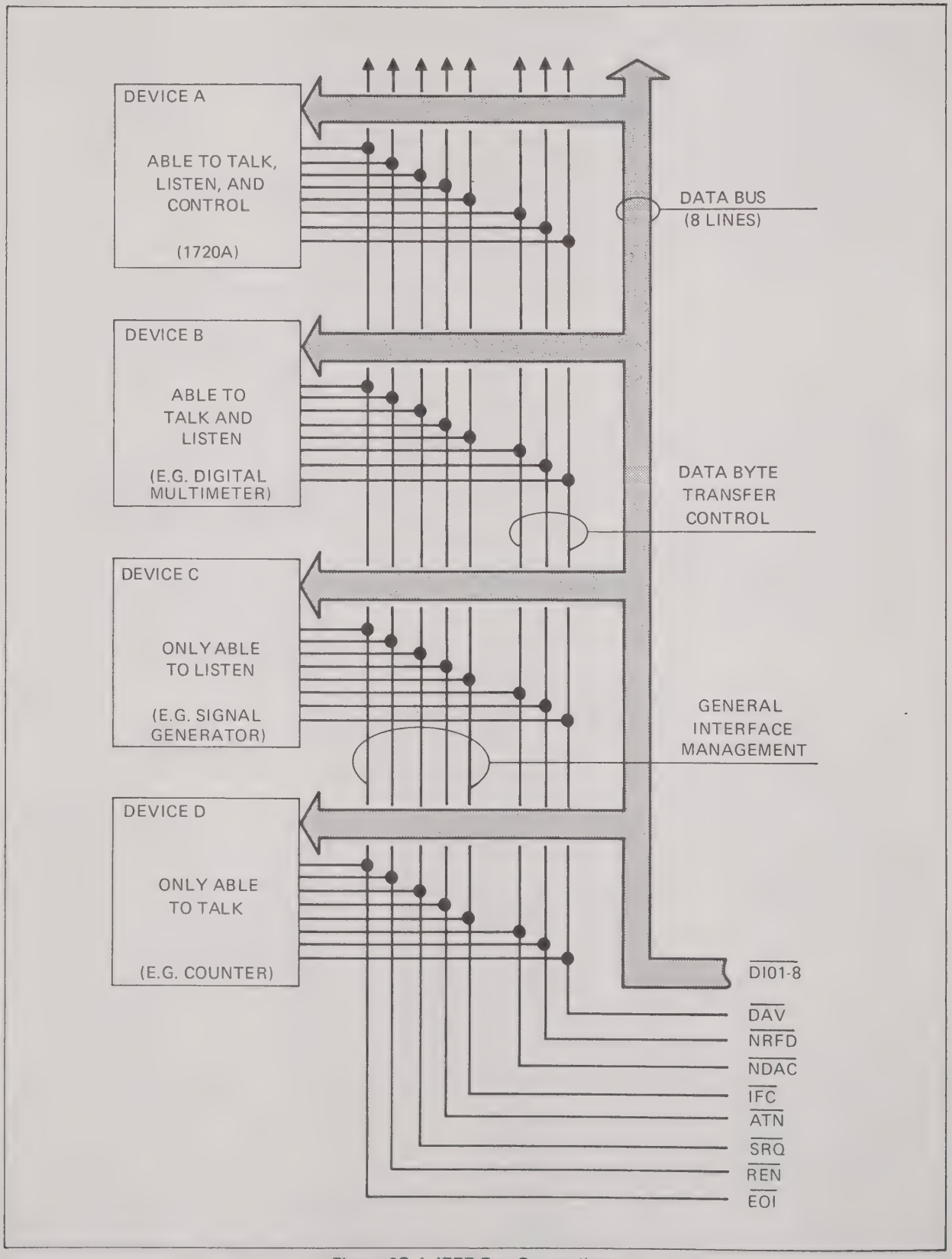


Figure 2C-1. IEEE Bus Connections

Table 2C-3. Command Message Categories

CATEGORY	DEFINITION
Primary Addressed Commands	Provide a contoller with the capability to control a selected group of devices. Since only those devices previously addressed to listen must respond to an addressed command, the command must be preceeded by the device address.
Universal Commands	Provide a controller with the capability to control all system devices. Since all devices must respond to a universal command (if capable), the command does not need to be preceeded by a device address.
Primary Address Command	Provide a controller with the capability to designate a device to function as a talker (source) or listener (acceptor).
Secondary Commands	Provide a controller with the capability to implement the extended talk or listen capabilities of a device, as well as the means to conduct a system parallel poll. A secondary command is the second byte of a two byte command.

2C-22. Operational Sequences

2C-23. Figure 2C-4 shows two possible minimum system configurations. One possible operational sequence, by a controller, for each of these systems is given in the following paragraphs.

2C-24. SYSTEM CONFIGURATION #1

1. The Controller initializes the interface devices by setting the IFC and REN lines true on each bus.

2. The Controller sends the listen address (MLA) of the voltmeter followed by the program data for the voltmeter on Bus #0.

3. The Controller sends the unlisten command (UNL) on Bus #0.

4. The Controller waits for an operator input or some other event defined by the user program.

5. The Controller sends the listen address (MLA) of the voltmeter, followed by the program code required or a GET (Group Execute Trigger) to initiate a measurement on Bus #0.

6. The Controller sends the unlisten command (UNL) on Bus #0, then proceedes to go on to other tasks as demanded by the user program.

7. Upon completion of its measurement, the voltmeter set the SRQ– line true.

8. The Controller sends the talk address (MTA) of the voltmeter on Bus #0, then uses the three-wire handshake to get the data bytes representing the voltmeters measurement.

9. The Controller sends the untalk command (UNT) on Bus #0.

10. The Controller processes the data, formats it, or does whatever the user program dictates.

11. The Controller sends the listen address (MLA) of the printer on Bus #1.

12. The Controller sends the processed data to the printer on Bus 1, using the three-wire handshake.

13. The Controller sends the unlisten command (UNL) on Bus #1.

2C-25. SYSTEM CONFIGURATION #2

1. The Controller initializes Bus #0 by setting the IFC and REN lines true.

2. The Controller sends the listen address of the voltmeter (MLA), followed by the program data for the voltmeter.

3. The Controller sends the unlisten command (UNL) on Bus #1.

4. The Controller waits for an operator input or some other event defined by the user program, then sends the listen address command (MLA) to the voltmeter, followed by the code that initiates the voltmeters reading on Bus #0.

5. The Controller sends the unlisten command (UNL) on Bus #0.

6. The Voltmeter makes its measurement and sets the SRQ– line true.

7. The Controller sends the listen address (MLA) of the printer and the talker address (MTA) of voltmeter.

8. The voltmeter uses the three-wire handshake to send the byte of data representing its measurement to the printer. This byte must be in a mutually understood code.

9. The Controller sends the unlisten (UNL) and untalk (UNT) commands on the bus.

CATAGORY	MNEMONIC	MESSAGE NAME	MANAGEMENT SIGNALS	INTERFACE FUNCTIONS REQUIRED TO:		DEVICE RESPONSE
				SEND	RECEIVE	
Primary Addressed Commands	GTL	Go To Local	ATN	C	RL	Device returns from remote to local mode of operation.
	SDC	Selected Device Clear	ATN	C	DC	Device configures to a pre-defined state defined by device manufactor.
	PPC	Parallel Poll	ATN	C	PP	Device Becomes addressed to configure so that if the next command is a PPE (Parallel Poll Enable) the device becomes configured to respond with a status bit when the controller initiates a parallel poll (EPI and ATN set) another next command reverts the device to be unconfigured.
	GET	Group Execute Trigger	ATN	C	DT	Initiates a device function among a selected (addressed) group of devices.
	TCT	Take Control	ATN	C	C	Device previously addressed to talk assumes Controller-In-Charge function and previous Controller-In-Charge becomes idle.
Universal Commands	LLO	Local LockOut	ATN, REN	C	RL	Disables Device Local controls so that devices can not prevent REN from putting devices in Remote Mode.
	DCL	Device ATN	ATN	C	DC	Devices configure to conditions pre-defined by their manufactors.
	PPU	Parallel Poll Unconfigure	ATN	C	PP	Devices return to the Parallel Poll unconfigured condition (no logic level or DIO line assigned for status bit response to Parallel Poll). If next command is PPE, device will reconfigure for Parallel Poll.
	SPE	Serial Poll	ATN	C	T, TE	Devices enter serial poll mode so that a subsequent talk command (MTA) will cause the device to write its status byte (see STB, RQS) on the DIO lines when the bus enters the data mode.
	SPD	Serial Poll Disable	ATN	C	T, TE	Devices exit serial poll mode and can no longer talk a status byte onto the DIO lines.
Primary Address Commands	MLA	My Listen Address	ATN	C	L, LE, RL, T	Device becomes addressed to listen if DIO 1-5 equals internal address switch setting unless a secondary address (MSA) is required. Device also enters Remote Mode if TEN is set. Device with capability will unaddress itself to talk. 31 addresses are available.
	UNL	UNListen	ATN	C	L, LE	Device becomes unaddressed to listen.

Table 2C-4. Command Bus Messages

CATAGORY	MNEMONIC	MESSAGE NAME	MANAGEMENT SIGNALS	INTERFACE FUNTIONS REQUIRED TO:		DEVICE RESPONSE
				SEND	RECIEVE	
Primary Address Commands (cont)	MTA	My Talk Address	ATN	C	T, TE L	Device becomes addressed to talk if DIO 1-5 equals internal address switch settings unless a secondary address (MSA) is required. Device with capability will unaddress itself to listen. 31 addresses are available
	OTA	Other Talk Address	ATN	C	T, TE	Devices become unaddressed to talk if any MTA other than their own is on DIO 1-5
	UNT	UNTalk	ATN	C		Devices become unaddressed to talk.
	MSA	My Secondary Address	ATN	C	TE, LE	Device with extended talker capability becomes unaddressed to talk if first talk address byte (MTA) equaled its address switch settings but this byte does not.
	OSA	Other Secondary Address	ATN	C	RE	Device which was prviously parallel poll configured by PPE command becomes unconfigured (no longer has status bit logic level and DIO line assigned). If the next command is PPE, the device will become reconfigured. This command must be preceded by the PPC command.
	PPD	Parallel Poll Disable	ATN	C	TE	Assigns a device a status bit logic level (high or low) and a DIO line to put the status on when IDY occurs (ATN and EOI concurrently). This command must be preceded by the PPC command.
	PPE	Parallel Poll Enable	ATN	C	PP	DIO Response Line # / Octal Code on Bus (if status bit is on) High / Low: 1: 140 150; 2: 141 151; 3: 142 152; 4: 143 153; 5: 144 154; 6: 145 155; 7: 146 156; 8: 147 157
Device Dependent	DAB	DAta Byte		T, TE	L, LE	Any byte on the DIO lines transferred between a source and one or more acceptors may represent program code, device status or measurement results.

Table 2C-4. Command Bus Messages (cont)

CATAGORY	MNEMONIC	MESSAGE NAME	MANAGEMENT SIGNALS	INTERFACE FUNTIONS REQUIRED TO:		DEVICE RESPONSE
				SEND	RECIEVE	
Device Dependent (cont)	EOS	End Os String		T, TE	L, LE	A source can write Line feed or Carrage Return or both in sequence on the DIO lines to provide printer control as well as to acknowledge the end of a string of data byte transfers.
	NUL	Null Byte		T, TE		Source writes zeros onto DIO 1-8 to initialize the bus.
	PPR 1-8	Parallel Poll Response		PP	C	Device status bit response on DIO line and of logic level assigned by PPE Command. This is the only data which is transferred without the 3-wire handshake.
	STB	Status Byte		T, TE	L, LE	Device reponse to serial poll STB is concurrent with ROS. Status Bits are on DIO 1-5
	RQS	Request		T, TE	L, LE	Device response to serial poll STB if device requested service by setting SRQ. RQS is concurrent with STB and consisters of DIO 7 being set. SRQ is cleared provided the condition causing it no longer exists.

Table 2C-4. Command Bus Messages (cont)

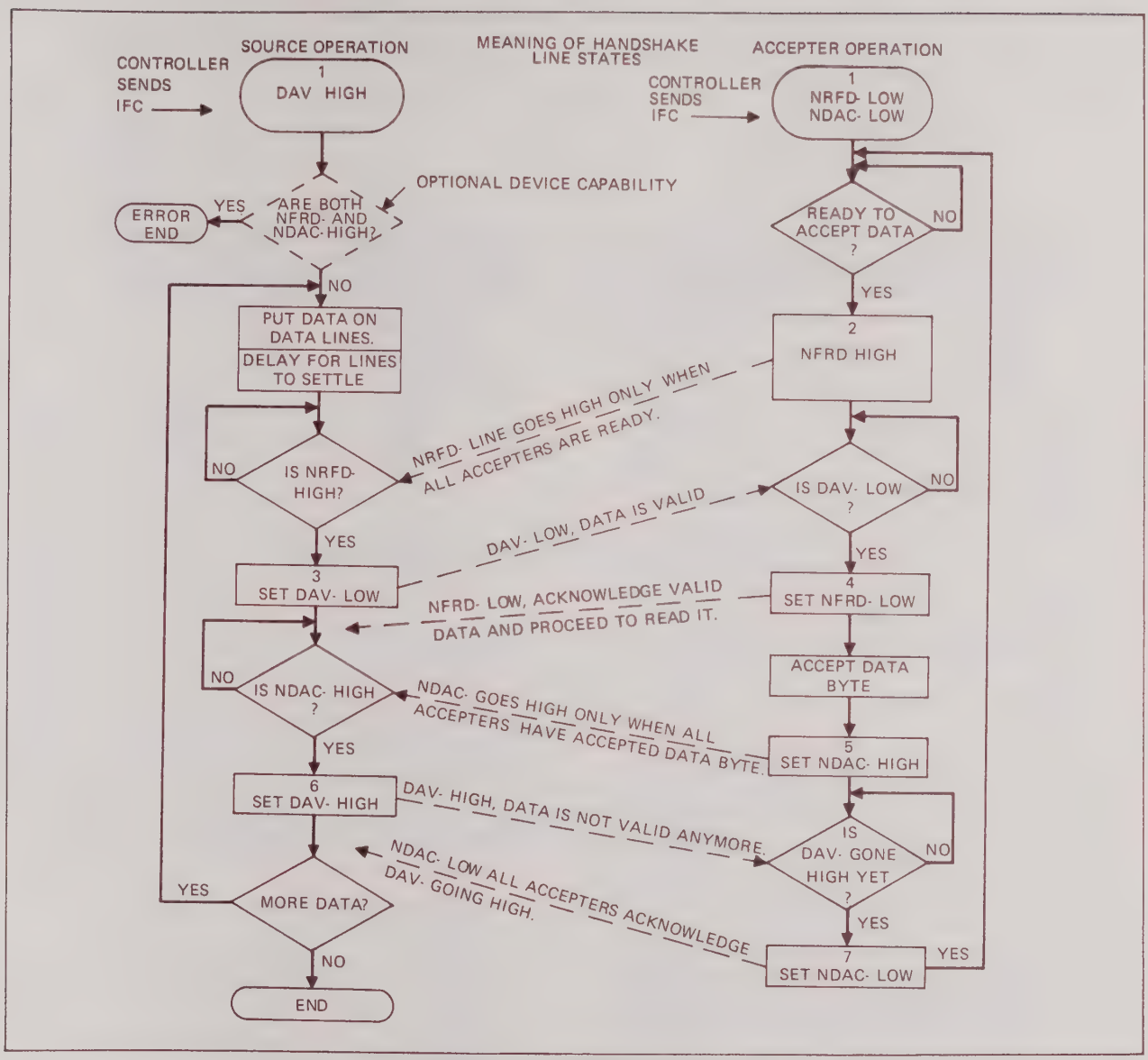


Figure 2C-2. Handshake Flowchart

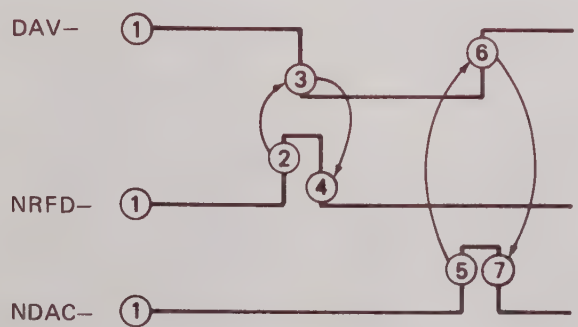


Figure 2C-3. Handshake Timing

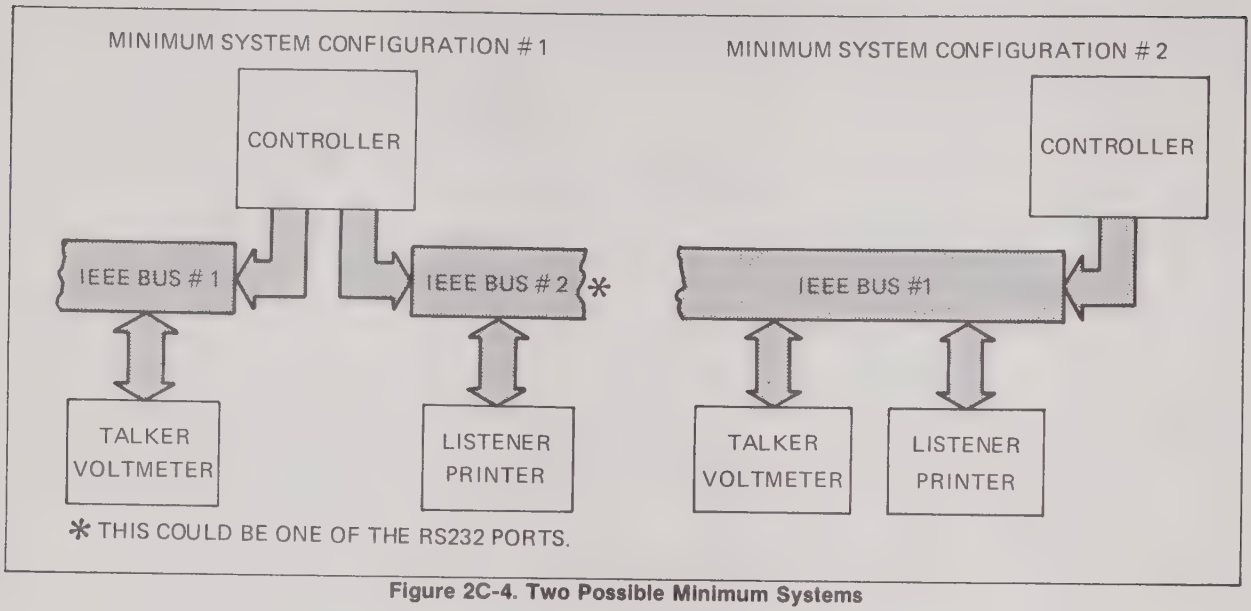


Figure 2C-4. Two Possible Minimum Systems

2C-26. IEEE-488 INTERFACE MODULE THEORY OF OPERATION

2C-27. Introduction

2C-28. The IEEE-488 Interface Module allows the 1720A Controller to communicate with two IEEE-488 buses. This allows the user to increase system throughput by configuring the system with fast instruments on one bus and slow instruments on the other. Since each bus has a capacity of 15 instruments, the 1720A can directly control 28 other devices. Each IEEE-488 Bus has the ability to act as non-system controller and to pass control (with appropriate software, not presently available) and to implement most of the interface functions within the IEEE Standard 488-1978, including all controller functions.

2C-29. In addition, selectable interrupt driven bus communications can free the 1720A for other tasks during slow bus transfers. Also, each bus interface has readback circuitry to talk to itself and, thus, diagnose the operation of the hardware via a self-test routine.

2C-30. Two Independent IEEE-488 Ports

2C-31. Refer to the IEEE-488 Module Block Diagram in Figure 2C-5 and note that the two IEEE-488 Ports have many functional blocks in common. The two ports become functionally separate from the point at which the Data and Control Registers are implemented. From that point on, there are registers and functional blocks that are duplicated in each port. One important point that may not be obvious is that the output from the Data/Control Register goes to two places. The Data portion goes directly to the Port Drivers and the Control portion goes to the Handshake Control.

2C-32. With two exceptions (see Automatic IFC and Automatic DCL) the CPU Module uses software control, via the CRU Bus, to accomplish all IEEE Bus data transfers through the registers. For the purposes of this software control, there are four registers:

1. Reg. 1 Port 1, Data/Control (16 bits)
2. Reg. 2 Port 0, Data/Control (16 bits)
3. Reg. 3 Port 1, Status/Control (8 bits)
4. Reg. 4 Port 0, Status/Control (8 bits)

2C-33. The CPU Module can set (write) each bit in these registers by using the CROUT and CRCLK lines, or it can sense (read) each bit by using the CRIN line. This gives each bit in these registers two meanings, one for the output from the CPU to the IEEE Module (write), and one for the input to the CPU from the IEEE Module (read). To provide this capacity, there is actually a pair of hardware registers in the IEEE Interface Module for each software register. Each pair of hardware registers appears as one software register because each CRU bit address actually accesses the same bit in twodifferent hardware registers. When the software does a write (CROUT) it accesses the bit in one register, and when it does a read (CRIN), it accesses the bit in the other register.

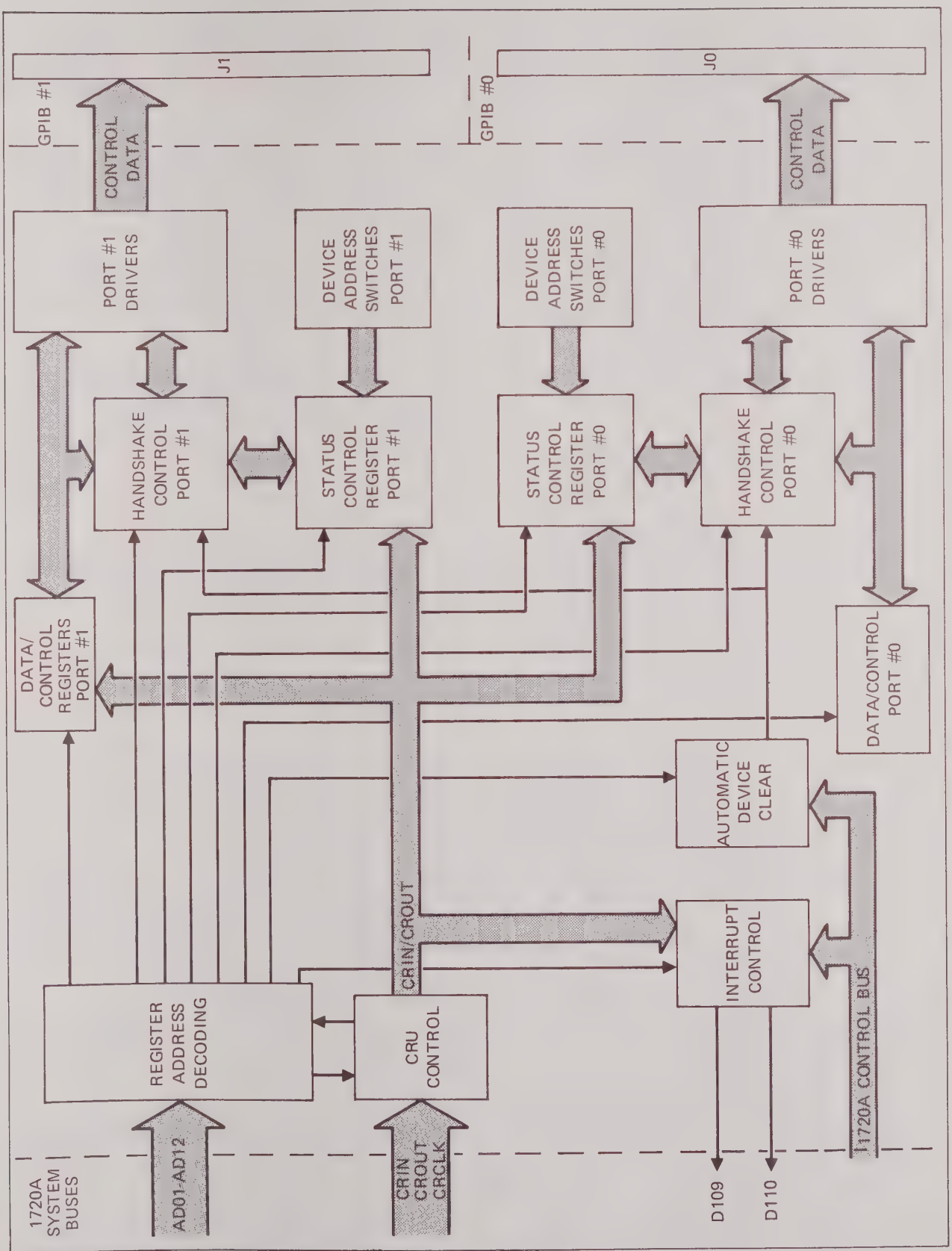


Figure 2C-5. IEEE Module Block Diagram

2C-34. The CPU Module addresses each bit in these registers with a separate CRU address. This address appears on the 1720A Address Bus bits AD01 through AD12. The Register Address Decoding on the IEEE Interface Module recognizes that the address is for one of the bits in one of the IEEE Interface registers and proceeds to decode AD04 through AD06 into one of six pairs of strobe/enable signals. One signal from this pair acts as a strobe to a 8-bit register that contains the bit that is accessed by the CROUT line (write) and is only active if the address is accompanied by CRCLK. The other signal from the pair acts as an enable to a 8-bit register that is accessed by the CRIN line (read). Table 2C-5 contains the IEEE Interface register address decoding and presents the above relationship in tabular form. The Chip # referred to in Table 2C-5 is the unit number of the 8-bit register on the IEEE-488 Interface Module schematic. The register number is the software register number referred to previously.

2C-35. Each 8-bit register that is strobed or enabled decodes Address Bus bits AD01, AD02 and AD03 into a bit number within the register. As stated previously, this bit can have two meanings, one for output to the CPU (read), and one for input from the CPU (write). Table 2C-5 shows this relationship in tabular form by giving the bit name for the write and read bits. These bit names are mnemonics which correspond with the IEEE Bus signals in some cases. In those cases where they do not, they reflect the function they have within the IEEE Interface Module. The following paragraphs use these mnemonics and give the full signal names which state the internal function of the signal.

2C-36. In summary, each IEEE-488 Interface circuit consists of a talker circuit and a listener circuit which complete the source handshake (SHS) and the acceptor handshake (AHS), respectively. Control and status bits in four registers dictate the operation of these handshakes, and the data bits control the data being exchanged. The CPU accesses these four registers via the CRU bus.

2C-37. IEEE Bus Read Operation

2C-38. The description that follows covers Port #1. The operation of Port #0 is the same. To read a byte of data from the IEEE Bus the software controlling the CPU goes through the following, or similar, steps:

1. Set LTN (Listen) or LA (Listen Always) to a 1. Setting the LA bit (Registers 3 and 4, bit 7) forces the associated IEEE Interface Port to listen to itself as well as other devices (sources) and respond accordingly. Two things happen:

   a. NRFD - on the IEEE bus goes false (high).

   b. The RDY interrupt is enabled if the RDY mask bit =1.

2. The RDY bit which goes to a 1 when DAV- on the IEEE Bus goes true (low).

   a. This can be done by a repetitive read of register 1 bit 11.

   b. This can be done with an RDY interrupt followed by a read of register 1 (if the RDY Mask bit is =1 to allow the interrupt to occur).

3. Read the incoming data byte from Register 1 when RDY goes to a 1. If the data byte is not the last byte (no terminating character or no EOI), then toggle ACC (set high, then low) and repeat the process (go back to step 1).

4. If the data byte is the last character, the user software may choose to hold NRFD on the IEEE Bus true (low, the bus is not ready) until the handshake is finished. To do this, the software:

   a. Sets RHO (Ready Hold Off) to a 1 to hold NRFD- true.

   b. Toggles ACC to assert NDAC- high (data accepted) and complete the handshake. The Handshake Control will set NDAC- low again.

   c. Tests HDN (Handshake Done) until true (= 1). This occurs when the current talker on the IEEE Bus puts DAV- high at the end of the handshake.

   d. Sets ATN in Register 1 to a 1 to make ATN- on the IEEE Bus true (low).

   e. Sets RHO to a 0 to set NFRD- false (high) on the IEEE Bus.

2C-39. Step 4 in the previous sequence is sometimes necessary to prevent some instruments from placing a new (and sometimes undefined) data byte on the bus and having it lost when the controller asserts ATN.

2C-40. IEEE Bus Write Operations

2C-41. To write a byte of data on the IEEE Bus the software controlling the CPU must go through the following, or similar, steps:

1. Set LTN to 0 to release control of the two listen handshake lines, NRFD- and NDAC-, and take control of the talker handshake line from the talker DAV-.

2. Write the data byte into register 1 bits 0 through 7 to place the data byte onto the DI01 through DI08 lines on the IEEE Bus.

Table 2C-5. IEEE Interface Register Address Decoding

REG. NAME	CHIP #	STROBE/ ENABLE	BIT #	ADDRESS BUS 6 5 4 3 2 1	HEX ADD	REG. #	BIT NAME WRITE	BIT NAME READ
Data Port #1	U6, Write U13, Read	WDATA1 RDATA1	0	0 0 0 0 0 0	000	1	D101	D1I1
			1	0 0 0 0 0 1	001		D102	D1I2
			2	0 0 0 0 1 0	002		D103	D1I3
			3	0 0 0 0 1 1	003		D104	D1I4
			4	0 0 0 1 0 0	004		D105	D1I5
			5	0 0 0 1 0 1	005		D106	D1I6
			6	0 0 0 1 1 0	006		D107	D1I7
			7	0 0 0 1 1 1	007		D108	D1I8
Control Port #1	U5, Write U12, Read	WCNT1 RCNT1	8	0 0 1 0 0 0	008		EOI1	EOI1R-
			9	0 0 1 0 0 1	009		RHO1-	SRQR1
			10	0 0 1 0 1 0	00A		LTN1	HDN1-
			11	0 0 1 0 1 1	00B		ACC1	RDY1
			12	0 0 1 1 0 0	00C		NBA1	DUN1
			13	0 0 1 1 0 1	00D		IFC1	NLR1-
			14	0 0 1 1 1 0	00E		REN1	RENR1-
			15	0 0 1 1 1 1	00F		ATN1	ATNR1
Data Port #0	U62, Write U55, Read	WDATA0 RDATA0	0	0 1 0 0 0 0	010	2	D101	D0I1
			1	0 1 0 0 0 1	011		D102	D0I2
			2	0 1 0 0 1 0	012		D103	D013
			3	0 1 0 0 1 1	013		D104	D0I4
			4	0 1 0 1 0 0	014		D105	D015
			5	0 1 0 1 0 1	015		D106	D016
			6	0 1 0 1 1 0	016		D107	D017
			7	0 1 0 1 1 1	017		D108	D0I8
Control Port #0	U61, Write U54, Read	WCNT0 RCNT0	8	0 1 1 0 0 0	018		EOI0	E0IR0-
			9	0 1 1 0 0 1	019		RH00	SRQR0-
			10	0 1 1 0 1 0	01A		LTN0	HDN0-
			11	0 1 1 0 1 1	01B		ACC0	RDY0
			12	0 1 1 1 0 0	01C		NBAD	DUNO
			13	0 1 1 1 0 1	01D		IFC0	NLR0-
			14	0 1 1 1 1 0	01E		RENO	RENR0-
			15	0 1 1 1 1 1	01F		ATN0	ATNR0
Status/Control Port #1	U4, Write U2, Read	WGP1 RGP1	0	1 0 0 0 0 0	020	3	INTMSK1	D1A0
			1	1 0 0 0 0 1	021		DUNMSK1	D1A1
			2	1 0 0 0 1 0	022		RDYMSK1	D1A2
			3	1 0 0 0 1 1	023		SRQMDK1	D1A3
			4	1 0 0 1 0 0	024		CIC1	D1A4
			5	1 0 0 1 1 0	025		SRQOUT1	SYNC1
			6	1 0 0 1 1 0	026		ATNMDK1	INT1
			7	1 0 0 1 1 1	027		LA1	IFCR1-
Status/Control Port #0	U60, Write U3, Read	WGP0 RGP0	0	1 1 0 0 0 0	030	4	INTMSK0	D0A0
			1	1 1 0 0 0 1	031		DUNMSK0	D0A1
			2	1 1 0 0 1 0	032		RDYMSK0	D0A2
			3	1 1 0 0 1 1	033		SRQMSK0	D0A3
			4	1 0 0 1 0 0	034		CIC0	D0A4
			5	1 1 0 1 0 1	035		SRQOUT0	SYNC0
			6	1 1 0 1 1 0	036		ATNMSK0	INTO
			7	1 1 0 1 1 1	037		LA0	IFCR0-

Note: Hex Add is for Board #0

3. As desired by the programmer, set E0I, ATN, REN, SRQ, or IFC at this time.

4. Toggle NBA (New Byte Available) high then low to make DAV- on the IEEE Bus go low, then high to signify a write operation. The Handshake Control block automatically controls the DAV- line levels to respond to NRFD- from the listeners. See Figure 2C-3 and 2C-4.

5. Test DUN until true (= 1).

a. This can be done by a continuous read of register 1.

b. This can be done by responding to a DUN interrupt with a read of register 1 (if the DUN Mask bit is =1 to allow the interrupt to occur).

DUN becomes true when all acceptors have taken the data byte (NDAC- goes high).

2C-42. Automatic IFC

2C-43. Both Ports send IFC (Interface Clear) upon power-up or power-fail if those ports have their SYC (System Controller) switch set. (Described later under Address Switch). The backplane signal DCOK signals power-up or power-fail to the IEEE Interface Module. Refer to the IEEE-488 Interface Schematics and follow the signal DCOK to U26 pin 10 in the Handshake Control Circuit. When DCOK goes low (DC not OK) the output at U26-8 goes high to enable one input to a NAND gate. The SYC bit enables the other input to this gate. As a result, IFC- goes true (low) on the IEEE Bus.

2C-44. Automatic DCL

2C-45. If the IEEE Bus interface circuits are configured as SYC (System Controller) and CIC (Controller In Charge) they will generate a DCL (Device Clear) on the IEEE Bus when certain backplane lines are true (note that CIC is bit 4 in registers 3 and 4). Refer to the Automatic Device Clear Block on both Figure 2C-5 and the Schematic. The backplane signals that cause DCL are:

1. HERR (Hardware Error), the result of a memory parity error detection.

2. ACOK (AC OK), a power warning from the power supply.

3. BCLR (Bus Clear), a CPY signal resulting from the execution of a RESET (RESET) instruction.

2C-46. Five Maskable Interrupts

2C-47. Refer to the Interrupt Control Block on both the Schematic and the Block Diagram, as this discussion proceeds. There are five interrupting sources available:

1. DUN - Occurs when all acceptors have taken data byte during a source handshake (NDAC- goes high)

2. RDY - Occurs when DAV- (Data Valid) on the IEEE Bus goes low (true) during an acceptor handshake.

3. SRQ - Occurs when SRQ- on the IEEE Bus goes low (true).

4. ATN - Occurs when ATN- on the IEEE Bus goes low (true).

5. IFC - Occurs when IFC- on the IEEE Bus goes low (true).

2C-48. Whenever one of these interrupts occurs, it sets the corresponding bit in registers 1 or 2 to a 1. These bits act as inputs to the Interrupt Control. Interrupt Control takes these inputs and ANDs them with the various MASK bits from Registers 3 and 4.

2C-49. The MASK bits in registers 3 and 4 can mask CUN, RDY, SRQ, and ATN individually or in any combination. The INT MASK bit masks all five interrupt sources at once and masks the IMT1 (Interrupt Board 1) or the INT2 (Interrupt Board 2) depending on which position the Board # Select Switch is in.

2C-50. The output of the Interrupt Control, INT1 or INT2, represents an ORing of all five interrupts. If any one of them occurs, and a corresponding mask bit is not set to a 0, INT1 or INT2 occurs. INT1 and INT2 are then ORed to produce the IEEE board interrupt to the processor. Hence, upon responding to an IEEE interrupt, the software must check to see if INT1 or INT2 is asserted.

2C-51. Configuration Switches

2C-52. Refer to the Dual IEEE-488 Module in Figure 2C-6. There are two configuration switches, one for Port 1 and one for Port 0. The individual switches are low true. Pushing a switch to an off position sets that position to a logical 1. Detail A on Figure 2C-6 shows a setting of 001001XX (X = don't care). This is equivalent to an address of 4 with SYC (System Controller) set to a 1.

2C-53. The software can read the setting of the address portion of these switches from registers 3 and 4 by reading bits 0 through 4. These bits are the DAx (Device Address) bits. The setting of the SYC bit can be read from bit 5 of registers 3 and 4. If desired, the programmer can use the device address as MTA (My Talk Address) for the Controller.

2C-54. Board # Select Switch

2C-55. Refer to the Dual IEEE-488 Module in Figure 2C-6. The Board # Select Switch serves two purposes within the IEEE Interface Module:

1. It changes the CRU address range that the registers occupy by making it necessary for AD07 to be set to a one if the switch is set to the Board #1 position and for AD07 to be set to a 0 if the switch is in the Board #0 position (AD01 through AD06 stay as in Table 2C-5).

2. It changes the Interrupt Control output to occur at DI10 if in Board #1 position and at DI09 if in Board #0 position. DI09 and DI010 are Data Input/Output lines that are low true to the Priority Interrupt circuit in the CPU during RINT cycles.

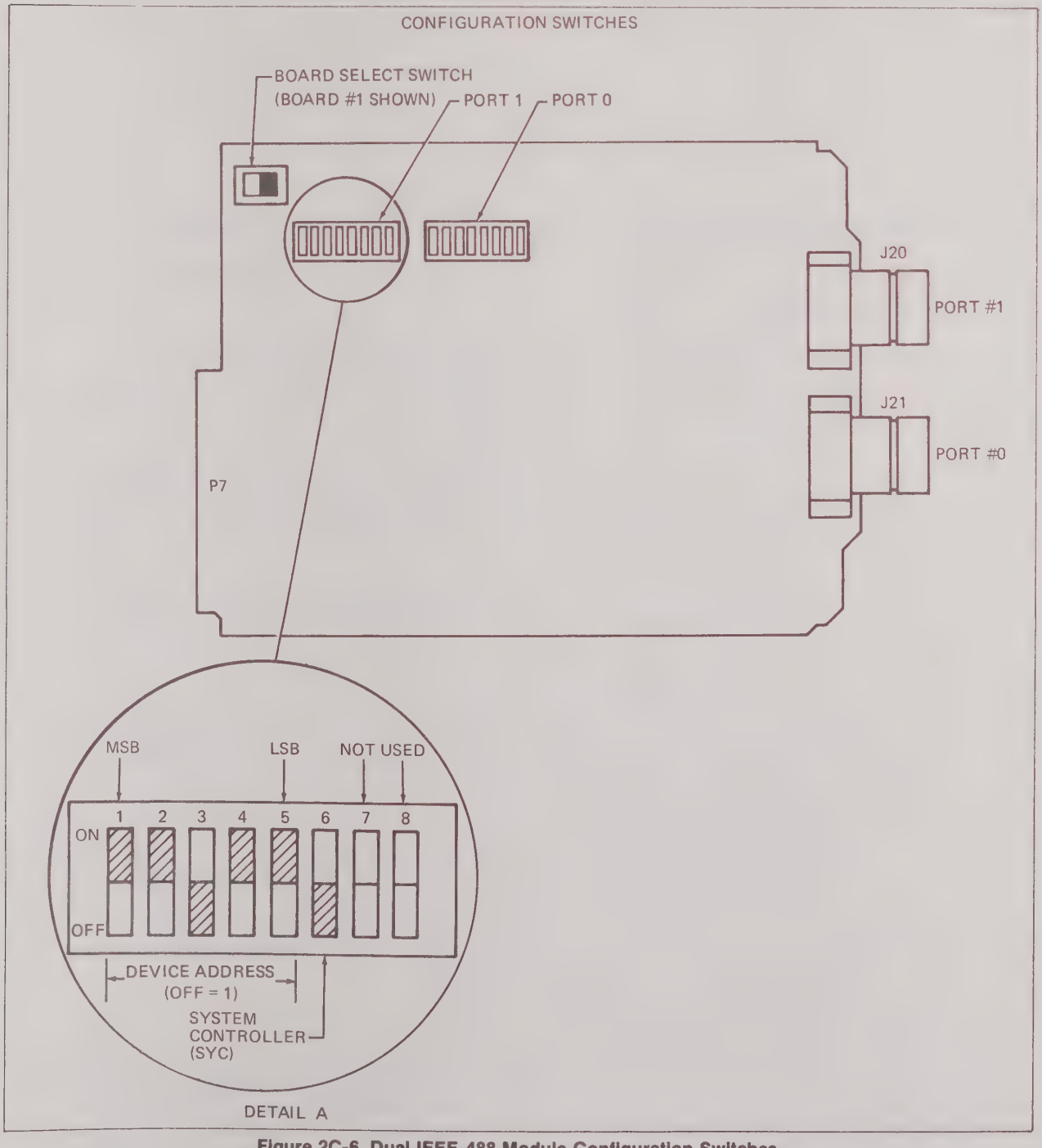


Figure 2C-6. Dual IEEE-488 Module Configuration Switches

Section 2D
Video/Keyboard Interface Module

2D-1. INTRODUCTION

2D-2. This Module performs three main functions within the 1720A Controller:

1. Receives display inputs from the CPU module, acknowledges their receipt, and stores them in a display memory.

2. Receives inputs from the Programmer Keyboard and the Touch-Sensitive Display then generates an interrupt to the CPU Module, and makes the information available to the CPU Module via the 1720A CRU Bus.

3. Generates the necessary outputs to enable the CRT Electronics to display the contents of the Display Memory.

2D-3. The module performs these functions with two major integrated circuit elements, a microprocessor and a CRT Controller. Figure 2D-1 shows the relationship between these two circuit elements and the other functional blocks in the module. The CRT Controller refreshes the CRT display via the video output and buffers the data from the display memory. The microprocessor interfaces the display memory to the 1720A System Buses and sets up the formatting of the CRT screen. It does this under the direction of a program that resides in ROM (Read Only Memory).

NOTE

This microprocessor and its CRU Bus, Address Bus, and Data Bus are separate and independent from the same components in the CPU Module. They do not leave the Video/Keyboard Module.

2D-4. FEATURES OF THE VIDEO/KEYBOARD MODULE

1. Four jumper-selectable cursor options:
   a. Blinking underline (standard)
   b. Blinking reverse video
   c. Non-blinking underline
   d. Non-blinking reverse video block
2. Two software selectable display formats:
   a. 16 lines of 80 characters (normal size characters)
   b. 8 lines of 40 characters (double size characters)
3. Software controlled keyboard lock
4. Four Special character modifiers:
   a. Highlight character(s)
   b. Blink character(s)
   c. Reverse video character(s)
   d. Underline character(s)
5. Eleven graphic symbols in double size.
6. Seven graphic symbols in normal size.
7. Characters are a 7 X 9 dot matrix set in a 8 X 14 cell.

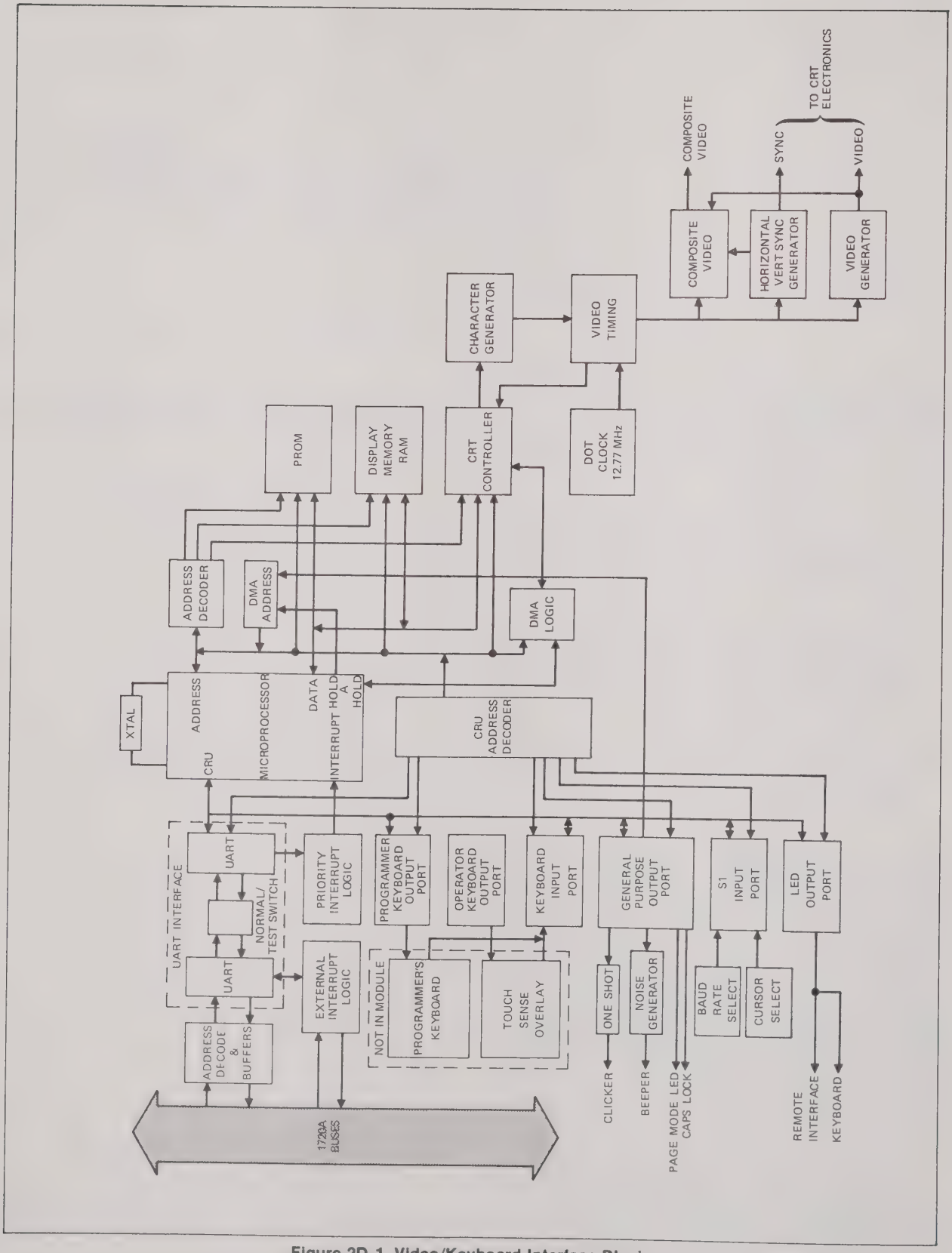


Figure 2D-1. Video/Keyboard Interface Block

2D-5. THE NORMAL/TEST SWITCH

2D-6. With S2 (NORMAL/TEST Switch) in the NORMAL position, the Video/Keyboard Module interfaces with the CPU Module via the 1720A CRU Bus. In general, the Video/Keyboard Module notifies the CPU Module when a keyboard entry on either the Programmer Keyboard or the Touch-Sensitive Display is made with an interrupt. The CPU Module then reads this character on its CRUIN line and, under software control, may display that character or take other action. If the CPU Module needs to display the character, it sends the character via the CRUOUT line to the Video/Keyboard Module for storage in a display memory and for display on the CRT. While S2 is in the NORMAL position, this is the only way characters can be displayed. This entire process is described in detail in the paragraphs that follow. Refer to the Test Mode Operation discussion for the differences in operation when S2 is in the TEST position.

2D-7. KEYBOARD SCANNING

2D-8. The microprocessor uses CRU lines to scan both the Programmer Keyboard and the Touch Sensitive Display through the Keyboard Output and Input Port. Figure 2D-1 shows this relationship. The difference between the CRU feature of this microprocessor and the CRU feature of the microprocessor in the CRU Module is that this microprocessor uses its LSB address line (VAD00) for its CRUOUT line instead of having a separate CROUT line. The CRUCLK and CRUIN lines function the same way in both microprocessors.

2D-9. The Keyboard Output Port (U2/U3) receives address bits A10, A11, and A12 directly into the three select inputs of U2 (8-bit addressable latch). The enable input for the latch comes from the CRU Address Decoder and results from the decoding of A3, A5, and A6 (port select). By assigning a binary weight to A10, A11, and A12, the microprocessor can set any of the eight latches in U2 to a 1 or 0 with the CRUOUT line. The process just described is the same for all the output ports in this Module. The outputs from this port serve two purposes:

1. The U2 outputs 0, 1, 2, and 3 become PKSO through PKS3 (Programmer Keyboard Strobe) and leave the Video/Keyboard Module to act as binaryy weighted inputs to a decoder in the Programmer Keyboard electronics. This decoder decodes the four PKS lines to enable one of eight column select lines. If any key on that column is depressed on the Keyboard, the code for that key appears on the KRO through KR7 (Keyboard Row) lines on the input to the Video/Keyboard Module. The program routine in the PROM scans the keyboard by incrementing the count on PKS0 through PKS3.

2. The U2 outputs 4, 5, 6, and 7 act as inputs to U3 (BCD to decimal decoder). U3 decodes these outputs to create 1 of 10 OKS (Operator Keyboard Scan) signal lines which leave the Video/Keyboard Module to scan the rouch-sensitive grid columns. If any switch on the touch-sensitive matrix is closed in the selected column, the code for that switch position appears on the KRO through KR7 inputs to the Video/Keyboard Module.

3. The ROM routine must generate the scan signals for both keyboards one at a time, over and over again. In addition, it causes the microprocessor to perform the following analysis:

a. A key must remain valid for a minimum time (contact de-bounce).

b. If more than one key is depressed, the output is inhibited (N-key lockout).

c. If more than one Touch-Sensitive Display key is depressed, only one location is sent out.

d. If a key is depressed for approximately 1 second, the output is repeated.

2D-10. ASCII CODE GENERATION

2D-11. The KRO through KR7 lines act as inputs to U5, a multiplexer. By placing the correct port address on A3, 5, and 6, the microprocessor enables U5, and the A10, 11, 12 lines select one of eight KR lines as the output of U5. This output acts as input to the microprocessor (CRUIN) enabling it to read the selected bit in the port. (The process just described is the same for all input ports in the Module.) The ROM program causes the microprocessor to read all eight KR lines one at a time and, if any valid character code (key depressed or grid switch closed) is present, the microprocessor converts the code for transmission to the CPU Module as follows:

1. If a key on the Programmer Keyboard is depressed, the output is the ASCII code of that key with the eighth bit equal to 0.

2. If a grid switch is closed on the Touch-Sensitive Display, the output is a binary number from 1 to 60 with the eighth bit equal to 1.

3. If a remote keyboard is implemented (both CTRL and SHIFT on) the output is a binary number from 61 to 124 with the eighth bit equal to 1. The available Remote Keyboard Cable does not allow 16 of the numbers in this range to be used.

2D-12. DISPLAY MEMORY

2D-13. The microprocessor stores the ASCII coded character received from the CPU module in a static RAM display memory. The position of the cursor determines the memory location. The Display Memory capacity is 2048 bytes. The microprocessor uses this capacity in a certain layout as shown in Figure 2D-2, Display Memory. The memory holds 16 rows of 128 bytes, as shown. The first 80 bytes in each row are the 80 characters to be displayed in one row of text on the CRT screen. The next 48 bytes in each memory row make up the scratchpad RAM for the microprocessor and can contain such things as workspace register files and buffers.

2D-14. CLICKER

2D-15. If the code received on the KR lines is for a Programmer Keyboard key, the ROM program produces an audible click inside the keyboard to provide feedback to the keyboard operator. To do this, the microprocessor sets bit 6 in the General Purpose Output Port U17 (8-bit addressable latch) to a 1. The latch output triggers a one-shot (U19) whose output leaves the Video/Keyboard Module as the Clicker signal to the Programmer Keyboard which has a transducer (solenoid) that makes the click sound.

2D-16. INTERRUPTS

2D-17. The external Interrupt circuit pulls the DI08 line low during RINT if there is a character ready to be sent to the CPU Module via the UART (Universal Asynchronous Receiver Transmitter) Interface. This serial interface is made up of two UARTs of the same type described for the RS-232-C Ports in the CPU Module section.

2D-18. INTERFACE TO THE 1720A BUSES

2D-19. The microprocessor in the Video/Keyboard Module uses the first UART (U28) to send a character to the second UART (U10). The CPU Module uses the second UART to send the character to itself, via the 1720A CRUIN bus line. The detailed sequence of events is as follows:

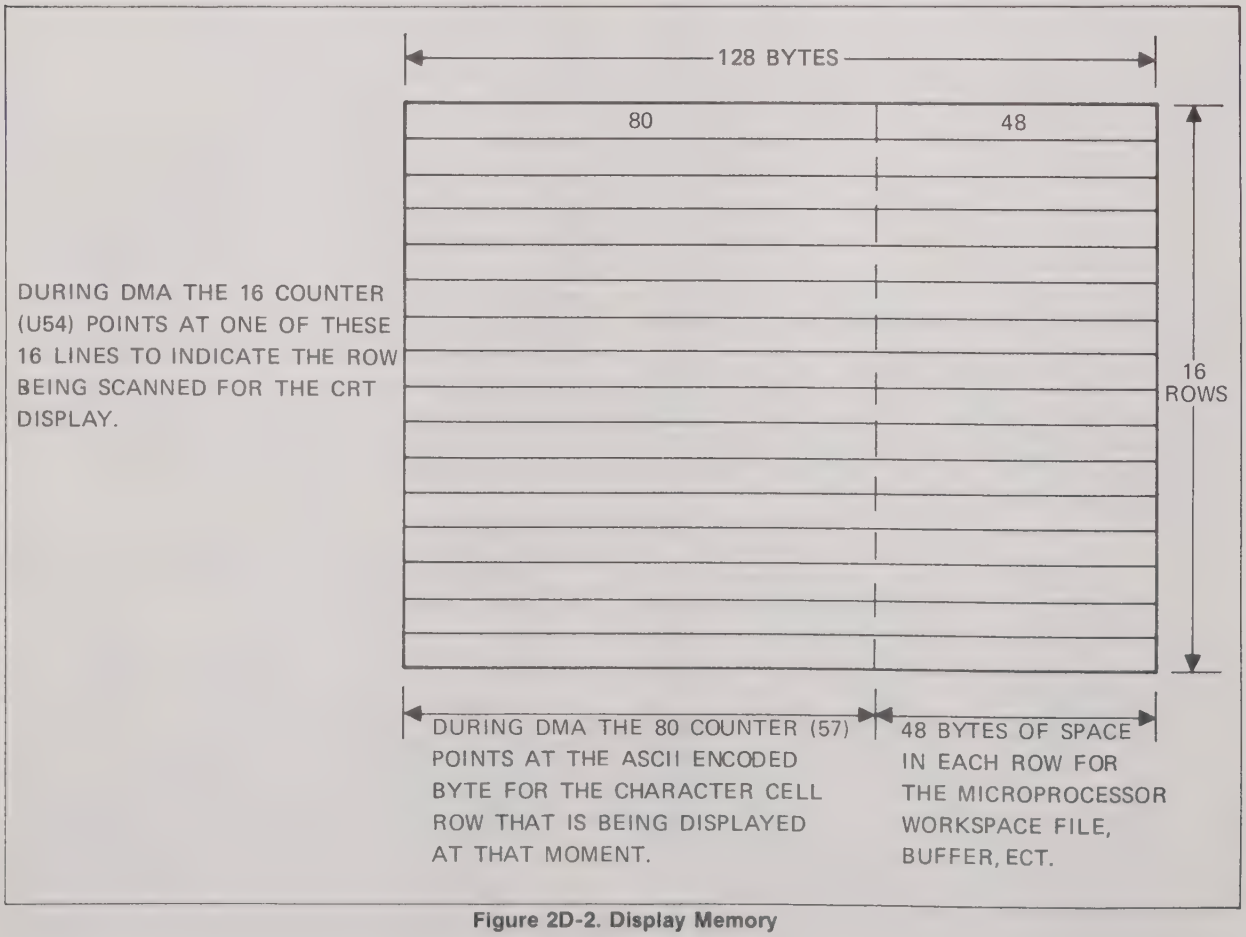

128 BYTES
80
48
16 ROWS
DURING DMA THE 16 COUNTER (U54) POINTS AT ONE OF THESE 16 LINES TO INDICATE THE ROW BEING SCANNED FOR THE CRT DISPLAY.
DURING DMA THE 80 COUNTER (57) POINTS AT THE ASCII ENCODED BYTE FOR THE CHARACTER CELL ROW THAT IS BEING DISPLAYED AT THAT MOMENT.
48 BYTES OF SPACE IN EACH ROW FOR THE MICROPROCESSOR WORKSPACE FILE, BUFFER, ECT.

Figure 2D-2. Display Memory

1. The microprocessor uses its CROUT line (VAD00 at CRUCLK time) to load the transmit buffer in U28 with the character that is ready to be sent. This would be a character from one of the keyboards.

2. The microprocessor initiates transmission of that character via XOUT on U28 to RIN on U10.

NOTE

The microprocessor can go on to other tasks after step 2.

3. When U10 has received the character, it generates an INT (Interrupt) output that puts a low level on the input of the gated line-driver U7.

4. The RINT signal from the CPU Module gates U7 on, and the output of U7 goes low to pull DI08 low.

5. As soon as the CPU Module can recognize this level of interrupt, it causes U10 to send the character on the 1720A Bus CRUIN line.

6. If the CPU Module wishes to send a character to the Video/Keyboard Module for either control purposes or for display, the reverse of the above occurs. U10 receives the characters from the CPU Module and U28 passes it on to the microprocessor. The microprocessor stores the characters sent to it for display in the Display Memory, as described earlier, and acknowledges the receipt of the input via the Serial Interface.

2D-20. DISPLAY FORMAT

2D-21. The microprocessor acting under the direction of inputs from the CPU Module and its own ROM program, establishes the display format by programming the CRT Controller. Keep in mind that the CRT display can be either in regular size characters or double size characters. The CRT Controller can be programmed to generate from 1 to 80 characters per row but generates either 80 or 40 characters in the 1720A. It can be programmed to generate from 1 to 64 rows per frame but generates 16 or 8 in the 1720A. In addition, the number of lines (horizontal sweeps) per character row can be programmed from 1 to 16 but is programmed for 14 in the 1720A. The dot width is a function of the Dot Clock frequency. Character width is a function of the Character Generator output (discussed later). The discussion that follows deals with the generation of regular size characters. Double size character generation is described later in this section.

2D-22. CRT CONTROLLER

2D-23. The CRT Controller has the responsibility of interfacing the CRT raster scan display to the microprocessor. Its primary function is to refresh the display by buffering the Display Memory information and to keep track of the display position on the screen. It has special features to generate the cursor, highlight characters, underline characters, reverse video, and supress video. In addition, it uses DMA (Direct Memory Access) to gather the characters from the Display Memory.

2D-24. DMA FOR DISPLAY

2D-25. The CRT Controller retrieves characters from the Display Memory, using DMA, on a row-by-row basis and stores them in one of two internal row buffers. While using one of these row buffers for creating the display, it is filling the other with the next row to be displayed. This process happens over and over again to keep the 16 rows of display memory on the CRT continuously refreshed. Any changes in the contents of the display memory show up the next time the CRT Controller refreshes that particular portion of the display.

2D-26. Before the start of a frame, the CRT Controller requests DMA by setting its DRQ (Direct Memory Access Request) high. This high acts as an input to the DMA Logic which produces a HOLD input to the microprocessor in synchronization with the Clock. The microprocessor responds by setting the SC (Scroll) output bits of the General Purpose Port to the address of the line that is to be at the top of frame.

NOTE

Refer to Figure 2D-2, Display Memory. This address can be the first line in the Display Memory layout, or it can be any other line. For example, during a long listing of a program from the 1720A Main Memory, when the text fills the entire 16 lines on the CRT, the top line on the CRT display is replaced by the second line and so on until the CRT display is 15 lines long on the screen. The next line of text in the listing is then written on the screen in the bottom line. This requires that, in the Display Memory, the address of the top line of the frame change (increment) for each new line of text in the listing, and that the new line of text be stored at the old top of screen address which is now the new last line of the display.

2D-27. The microprocessor then causes the Address Decoder block output CKOF– to go low by setting A0=1 and A1=0 and bringing the CRUCLK output high. This enables U24 (a decoder). The CKOF– output goes low and sets U57 (80-counter) to a count of 0 and loads the SC0, SC1, SC2, and SC3 count into U59 (16 counter).

2D-28. Next the microporcessor floats its address and data lines and causes its HOLDA output to go high. HOLDA causes several things to occur:

1. The output of the 80 counter and the 16 counters (U57 and U54) become the memory address.

2. The DACK (DMA Acknowledge) input to the CRT Controller goes true.

3. The data from the first memory location of the line that is to be the top of the frame appears on the D0 through D7 lines.

4. The CRT Controller stores the character in one of its row buffers.

5. The CRT Controller continues generating DMA requests until the row buffer is filled. Each time HOLDA from the microprocessor goes true, the 80 counter increments. Every 80 counts, the 16 counter increments. Thus, the 16 counter contains the line number (0 through 15) and the 80 counter contains the character number (1 through 80) of the character within the line.

2D-29. CHARACTER GENERATION

2D-30. As soon as the first row buffer is full, the CRT Controller begins to send characters from the beginning of the buffer tothe Character Generator and simutaneously begins to fill the second row buffer. The character Generator has two inputs:

1. The CC0 through CC6 character code outputs of the CRT Controller.

2. The output of a counter (U35) that increments once for each HRTC (Horizontal Retrace) output of the CRT Controller.

NOTE

Refer to Figure 2D-3, Character Cell. The character that is on the CRT display occupies a 7X9 dot matrix in a character cell that is an 8X14 matrix. The top line in the character field is always blank to provide a space between characters on different rows. The output of U36 goes to all zeros (blank line) when the RS inputs go to all ones (count of 15 from U35). This occurs at the first line of each character when U35 gets an active (low) LD (Load) input. The LD input to U35 goes active (low) at the first horizontal line in the row due to the VTRC (Vertical Retrace) output of the CRT Controller. Thereafter, it goes low every time U35 reaches a count of twelve which occurs every 14 horizontal lines (see the RS Input To U36 on Figure 2D-3).

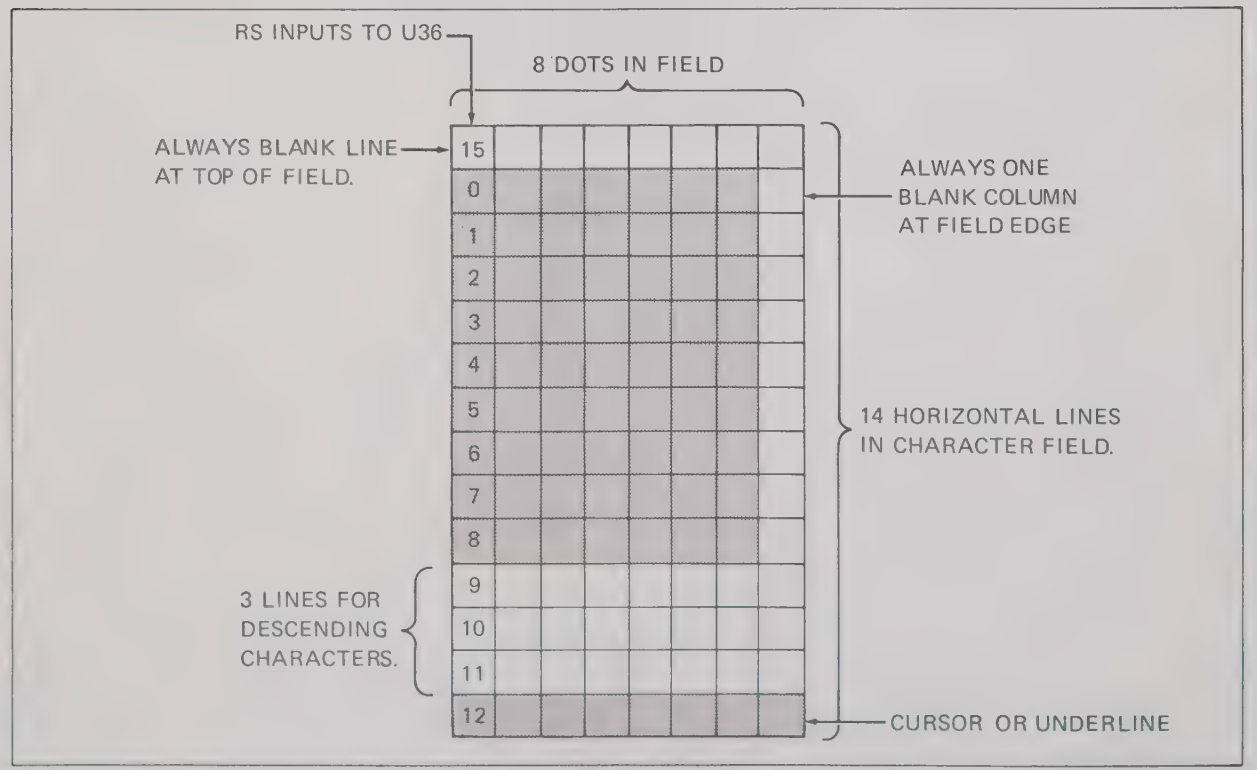


Figure 2D-3. Character Cell

2D-31. The Character Generator knows which CRT horizontal line within the character row (determined by the output of U35) it is working on, and it knows which character is has to display (CC0-CC6). The CC0-CC6 output of the CRT Controller is actually a memory address for the ROM of the Character Generator ROM. The ROM contents produce a pattern of seven ones and zeros for that particular line in that particular character. This pattern is the D0-D6 outputs of U36. The D0-D6 outputs act as inputs to a two to one multiplexer (U37/U38).

2D-32. The other inputs to the multiplexer provide graphic characters (generated by the CRT Controller). Decoding the LA0 and LA1 (Line Attribute Outputs) provides specific patterns. The use of graphic characters directly from the CRT Controller allows a graphic character to occupy areas of the cell outside the 7X9 matrix. This makes continuous lines between character cells possible.

2D-33. The 7-bit pattern from the multiplexer (U37 and U38) provides a parallel input to a shift register (U39). The shift register shifts the bits out serially at the Dot Clock rate to produce the video output of the Module. The Dot Clock rate determines the display dot width. The various modifiers from the CRT Controller, such as High Light or Reverse Video, control the CRT presentation of these dots. This process continues for each character in the row, as shown in Figure 2D-4. Each HRTC (Horizontal Retrace) increments the RS inputs to the Character Generator until all 14 horizontal lines in the character row are on the display.

2D-34. DISPLAY REFRESH AND SCROLLING

2D-35. As soon as the display of a row is complete, the CRT Controller reverses the roles of the two row buffers and begins to display the next row in the Display Memory. This process continues until the complete frame is on the CRT (16 rows). As soon as this is done, the entire frame-display process repeats, going to the Display Memory addresscontained in the 16 counter. This address should now be back at the original address of the top line in the display. This repetition of the same rows over and over again refreshes the CRT display.

2D-36. The microprocessor may also scroll the display. It does this by incrementing the scroll address, setting the 16 counter to this new value, and re-writing the previous scroll address Display Memory contents with a new line of text. The line pointed to by the new scroll address then appears at the top of the CRT Display, and the new line of text appears at the bottom of the CRT display. In either case, the contents of the Display Memory can be changed to change the CRT Display.

2D-37. CURSOR/BAUD RATE

2D-38. As stated in the introduction, the CRT Controller is capable of generating four types of cursor. It can place this cursor at any character location on the display by programming the CRT Controller. This is done by writing the appropriate information into a cursor row register and a character position register within the CRT Controller. The CRT Controller does the rest. In addition, the microprocessor reads the settings in S1 (positions A and B) from the S1 Input Port and programs the CRT Controller for the appropriate cursor. Refer to Table 2D-1.

2D-39. The microprocessor obtains the baud rate for data transmission by reading S1 switch settings C, D, and E from the RS-232 Port (P1) on the module. This Port is not used in the 1720A. The various jumper meanings appear in Table 2D-1.

2D-40. VIDEO

2D-41. This discussion on video is limited to the display of regular size characters. See Figure 2D-3, Character Cell.

2D-42. The purpose of the video outputs (VIDEO, HORIZ and VERT) is to produce a row of 80 cells across the CRT screen and make a display of 16 rows. Keep in mind as this discussion proceeds that the CRT Display Electronics produce a free running horizontal sweep from left to right that scans downward on successive sweep lines. A horizontal retrace occurs at the end of each line. When the sweep reaches the bottom of the screen it does a vertical retrace to the top of the screen and repeats the process. The HORIZ and VERT outputs of the Video/Keyboard Module, therefore, do not have to generate the sweep. Instead, they synchronize the sweep with the VIDEO output which intensity modulates the trace into a series of dots and spaces. The Video Suppress output of the CRT Controller suppresses the trace during the horizontal and vertical retrace times.

2D-43. Figure 2D-5 shows the four video signals under discussion and gives timing details for each. Use the following information to analyze Figure 2D-5:

1. The Video Suppress pulse (U40, pin 12) occurs once per row at the end of each row. It is 20 Character Clock pulses wide (to accomodate horizontal retrace) except at the end of the 16th row where it is 42 horizontal line periods wide (to accomodate vertical retrace).

2. The Horizontal sync pulse (HORIZ) occurs after the start of the Video-Suppress pulse (2.3 uS delay determined by U51 and has a width of 4.5 uS (determined by U52). It has a rate of 15960 Hz.

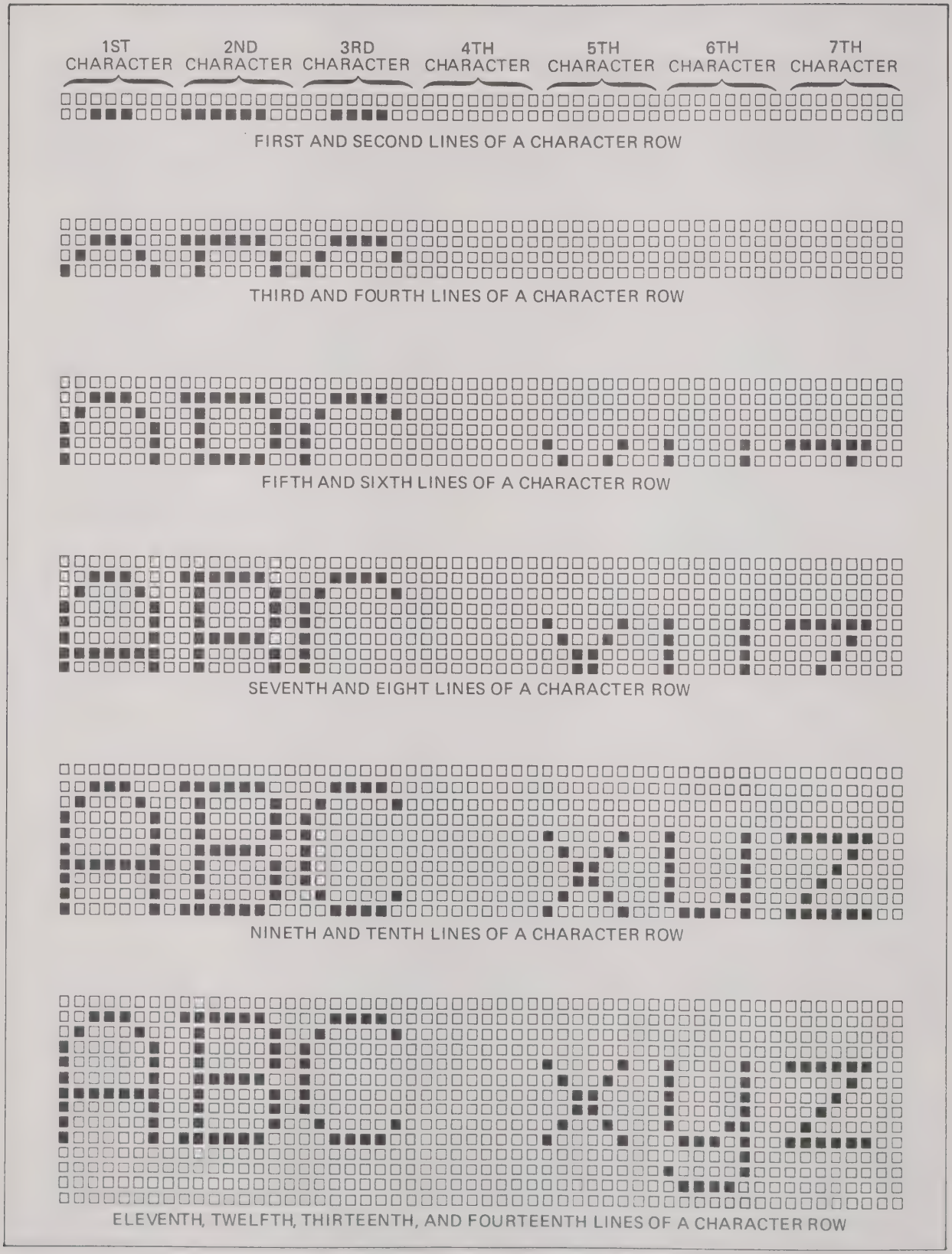

1ST CHARACTER
2ND CHARACTER
3RD CHARACTER
4TH CHARACTER
5TH CHARACTER
6TH CHARACTER
7TH CHARACTER
FIRST AND SECOND LINES OF A CHARACTER ROW
THIRD AND FOURTH LINES OF A CHARACTER ROW
FIFTH AND SIXTH LINES OF A CHARACTER ROW
SEVENTH AND EIGHT LINES OF A CHARACTER ROW
NINETH AND TENTH LINES OF A CHARACTER ROW
ELEVENTH, TWELFTH, THIRTEENTH, AND FOURTEENTH LINES OF A CHARACTER ROW

Figure 2D-4. Display of Charater Row

Table 2D-1. Cursor/Baud Rate Jumpers

A	B	C	D	E	
		•	•	•	Cursor = blinking reverse video block
X		•	•	•	Cursor = non-blinking reverse video block
	X	•	•	•	Cursor = blinking underline
X	X	•	•	•	Cursor = non-blinking underline
•	•				Baud rate = 9600
•	•	X			Baud rate = 4800
•	•		X		Baud rate = 2400
•	•	X	X		Baud rate = 1200
•	•			X	Baud rate = 600
•	•	X		X	Baud rate = 300
•	•		X	X	Baud rate = 150
•	•	X	X	X	Baud rate = 110

X = jumper
• = don't care

Note: The baud rate must be 4800 to communicate with the 1720A Bus.

3. The Vertical sync pulse (VERT) occurs at the end of the 16th row, and is delayed by 19 horizontal line periods (1.1 ms determined by U51). It isthree horizontal line periods wide (190us determined by U52) and has a rate of approximately 60 Hz. The display is not interlaced.

4. There is one video pulse for each dot on the CRT, and the absence of a pulse causes a blank space that is one dot wide. When using the character ROM (U36), the eighth pulse in any given character cell is always a blank (missing). This creates the blank space at the edge of each cell which makes the space between adjacent characters on the CRT display. There are 80 character cells of 8 pulses for each trace. The pattern of pulses and blanks in these 80 cells determines the dots and blanks in that horizontal trace. It takes 14 horizontal traces to build up one row of characters on the CRT display.

2D-44. Figure 2D-4, Display of a Character Row, shows how the 14 horizontal traces build up a hypothetical row of seven characters on the CRT display. Remember that it takes 16 rows of 80 characters to complete one CRT display of normal size characters.

2D-45. The ground on pin-14 of U38 produces the one blank dot per field. The output of U38 provides part of the parallel input to the VIDEO output at the Dot Clock rate. The dot clock output (U18, pin-8) determines the pulse width which determines the dot width on the screen.

2D-46. The video suppression output of the CRT Controller is used to suppress the horizontal retrace at the end of each horizontal line and the vertical retrace at the end of the 16th line. This high level output acts as the D-input to the quadruple D-type latch (U40). The high level produced on the Q3 output by this signal disables the blanking gate (U63) to blank the trace. This output also provides display blinking capability.

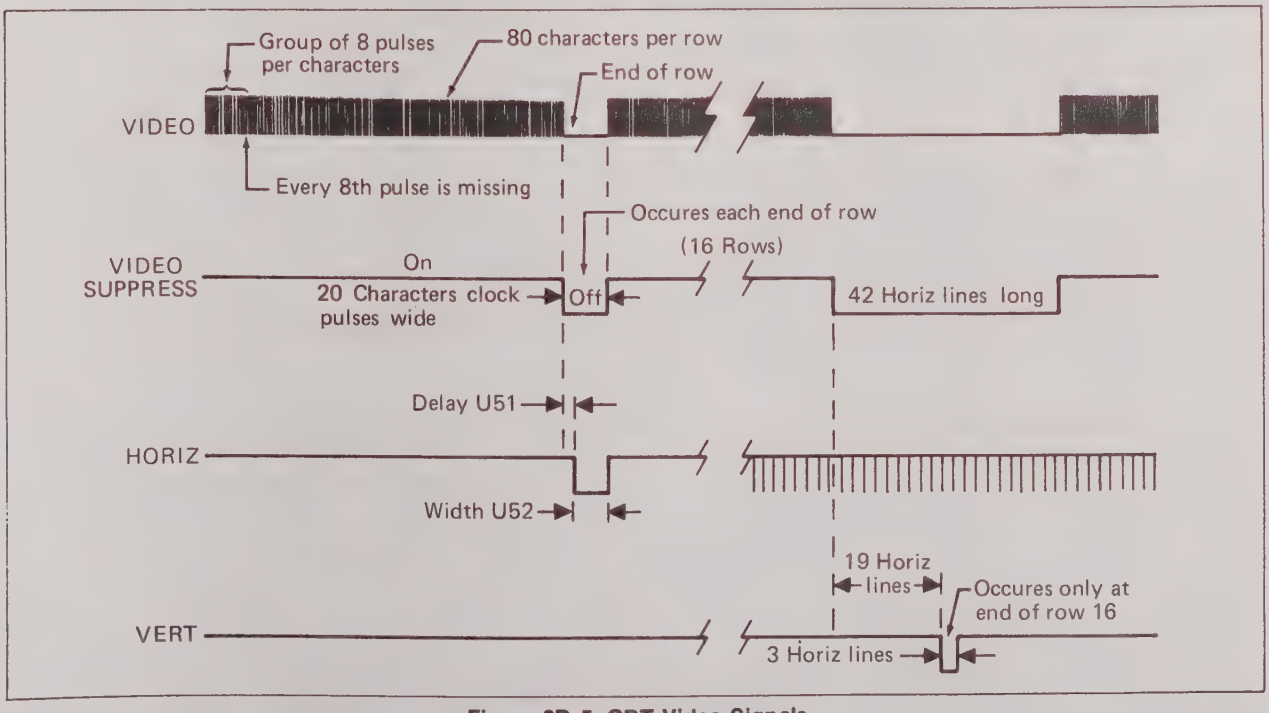


Figure 2D-5. CRT Video Signals

2D-47. The load input for U40 comes from an 8-bit counter (U22) which counts the Dot Clock output. As a result, U40 gets loaded with new video control data at the end of every character. Therefore, the latched video suppress output (pin 12) is always a multiple of eight dot Clock pulses (one character). At the end of the last character in a row, it is 20 characters (160 Dot Clock pulses) long. At the end of the 16th row, it is three character rows (42 horizontal lines) long.

2D-48. DOUBLE SIZE CHARACTERS

2D-49. To get double size characters, the microprocessor sets bit 4 in the General Purpose Output Port to a 1 (high). This causes two events to occur:

1. The output of a divide-by-two counter (U16, pin-9) begins to pass through U23 pin-3 because of the high level on pin-1. This causes U35 to count every other HRTC (divide-by-two). This makes each dot on the CRT display two horizontal lines tall instead of one.

2. The high level on pin-8 of U34 disables that NAND gate to block the full frequency Dot Clock from reaching OR gate U18 pin-10. As a result, only the 1/2 frequency Dot Clock from the divide by two output of U21 pin-5 reaches the clock input of U39. As mentioned earlier, the Dot Clock rate on this shift register determines the display dot width. The 1/2 rate makes the dots twice as wide and increments the character address (via CCLK on the CRT Controller) only 40 times per trace.

2D-50. In addition to the hardware considerations mentioned in the previous paragraph, the software must load the Display Memory for double size characters. This is done by duplicating the first 40 character positions in each row in memory and by using only eight rows of memory. The repeated horizontal trace scans the second set of 40 characters.

2D-51. BEEPER

2D-52. To create the beeper tone, the microprocessor, under the direction of the user software, places the correct code for CKON- on its A0 and A1 lines (A0=0 A1=1) and produces CRUCLK to enable decoder U24. The CKON- output of U24 goes low to trigger the one shot U19. The output of the one-shot turns on the tone generator (U64) for the duration of the pulse. The output of U64 drives a transducer to make the beep tone.

2D-53. LEDS

2D-54. The LED #1 through LED #7 outputs of the Video/Keyboard Module are the outputs of the 8-bit addressable latch U4. The microprocessor, under software direction, can set each of these outputs to either a one (on) or a zero (off). LED #1 through LED #5 go to the front panel receptacle from the Programmer Keyboard and LED #6 and 7 go to the Remote Interface. Keep in mind that these outputs are latched TTL levels (open collector) and as such could be used for a variety of purposes besides lighting the keyboard LEDs. When combined with the four PKS inputs, they make up an I/O Port to the 1720A that is available at the front panel for test or control purposes.

2D-55. CAPS LOCK LED

2D-56. The Caps Lock LED in the Programmer Keyboard receives a drive signal from the Video/Keyboard Module under software control in response to a caps lock key depression. The microprocessor produces this signal by setting bit 5 in the General Purpose Output Port to a one. When the module is first initialized, this LED is on.

2D-57. PAGE MODE LED

2D-58. The Page Mode LED receives a drive signal from the Video/Keyboard Module under software control in response to a page mode key depression. The microprocessor produces this signal by setting bit-7 in the General Purpose Output Port to a one. When the Module is first initialized, this LED is off.

2D-59. COMPOSITE VIDEO

2D-60. The Composite Video output is a 1 volt peak-to-peak signal into a 75 ohm load. This signal is easily verified at J24 (Phono Jack). Since it is a combination of all three video outputs, a voltage check is a quick way to verify its presence.

2D-61. TEST MODE OPERATION

2D-62. With S2 (NORMAL/TEST Switch) in the TEST position, the Video/Keyboard Module does not interface with the CPU Module via the 1720A CRU Bus. In this mode of operation, the microprocessor operates as usual, using the same ROM program as before. However, because S2 changes two signal paths, the results are different. Notice that with S2 in the test position, the XOUT output of U28 goes to the RIN input of U28 instead of going to U10. Notice also, that the RTS output of U28 goes to the CTS input of U28 instead of going to U10. These two changes effectively eliminate U10 and its interface to the CPU Module from the operation of the

system and allow the Video/Keyboard Module to display characters on the CRT directly.

2D-63. This direct display of characters begins when a key is pressed. The microprocessor proceeds to load the transmit buffer of U28, just as before. Now, however, U28 immediately loads its own receiver buffer instead of sending the character to the receiver buffer of U10. As soon as this buffer is full, U28 interrupts the microprocessor via the UINT- line. The microprocessor proceeds, under the ROM program control, to read the character in on its CRUIN line. When this operation is complete, the microprocessor stores the character in the display memory. The character then gets displayed just as if the character had come from the CPU Module. Note that the CTRL key modifier does not function normally because there is no software to generate a display code from two keys that are pressed in sequence but meant to be handled as if they were one key.

Section 2E
DMA-Floppy Module

2E-1. INTRODUCTION

2E-2. This discussion divides the DMA-Floppy Module into two functional circuit groupings:

1. DMA Controller and associated circuits.

2. Floppy Disk Controller (FDC) and associated circuits.

2E-3. These two circuit groups work together, under the direction of the CPU Module, to write and read data blocks on the built-in Floppy Disk drive unit. This process uses DMA (Direct Memory Access) to transfer the data to and from the Memory Module. In operation, the DMA-Floppy Module responds to memory mapped I/O operations by the CPU Module to receive address and control information. The CPU Module loads this address and control information into 12 different registers within the DMA-Floppy Module to control the various Module functions. After receiving this address and control information, the DMA-Floppy Module is capable of handling an entire DMA block transfer on its own.

2E-4. DMA CONTROLLER GENERAL DESCRIPTION

2E-5. Refer to Figure 2E-1. Notice the DMA Controller. It contains eight addressable registers and controls the byte mode DMA data transfers, with handshake, between the Floppy Disk Controller and the Memory Module. The following paragraphs describe the operation of the DMA Controller. Because the overall operation of the DMA process may not be apparent in all this data, it may be necessary to look ahead to the two topics that present the DMA operational flow (Module-To-Memory and Memory-To-Module DMA Transfers) and refer back to the details as necessary.

2E-6. Table 2E-1 presents the address, name and a brief description of all the registers in the DMA-Floppy Module. Look at the eight registers associated with the DMA Controller. A detailed description of each register appears in the following text. For now, note the function of these eight registers and remember that the CPU Module can use the register address to access each register with the Memory Mapped I/O process that was discussed in the CPU Module theory.

2E-7. The DMA Control Register (CR) holds the control command from the CPU Module. Each bit has a specific use, within the DMA Controller, as shown in Table 2E-2. Refer to the Schematic to locate specific signals.

2E-8. The Transfer Count Registers (TCR) are two 8-bit registers that hold the twos complement of the transfer count (in bytes) for DMA transfer operations. The DMA Controller increments the count by one after every DMA transfer. When the count reaches zero, bit 3 in the Status Register goes to a 1. If bit 3 in the Control Register is also a 1 then INTR goes low also, providing the STOPR- (STOP Request-) input to the IC is also high. Note that the STOPR- input is the 1720A ControlBus signal RQIN- (refer to the CPU Module theory.

2E-9. The Memory Address Registers (MAR) are two 8-bit and one 2-bit registers. The upper two bits (AD16 and AD17) need bit 6 in the Control Register set to a 1 before they are active. If bit 6 in the Control register is set to a one, then bit 16 in the Memory Address Register works as a carry from bit 15. The DMA Controller increments the contents of the Memory Address Register by one after every DMA transfer.

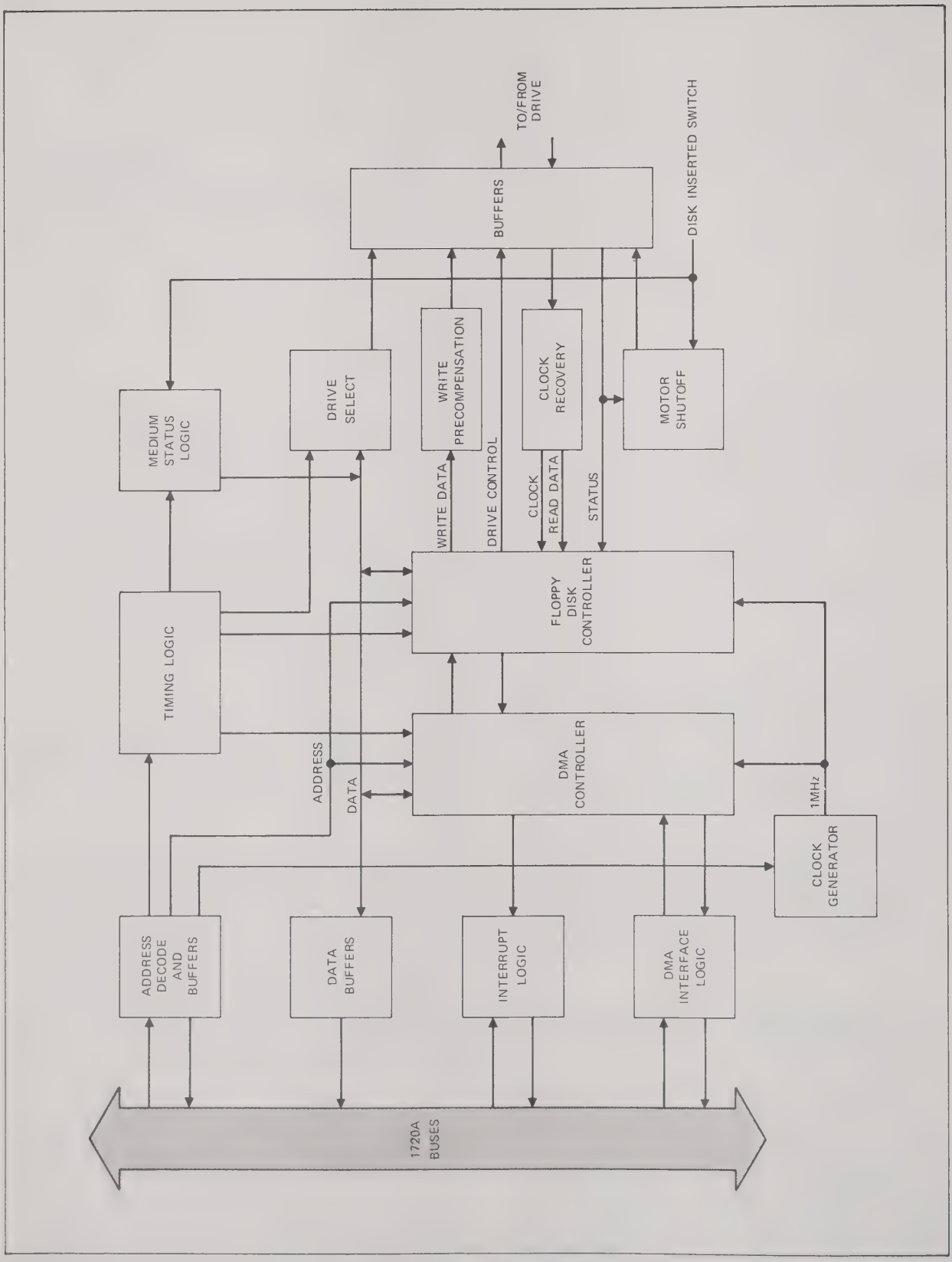


Figure 2E-1. Floppy Block Diagram

Table 2E-1. Register Address Assignments

ADDRESS	REGISTER	FUNCTION
3F3E0	STATUS/COMMAND	Floppy Status/Command
3F3E2	TRACK REG	Floppy current head position
3F3E4	SECTOR REG	Floppy address of desired sector position
3F3E6	DATA REG	Floppy parallel 8-bit read/write holding register
3F3E8	DRIVE UNIT SELECT/ MEDIUM STATUS	Disk drive unit select/ Medium Status
3F3F0	DMA CONTROL REG	8-bit DMA Controller control register
3F3F2	DMA STATUS REG	DMA Controller Status register
3F3F4	DMA TRANSFER COUNT Lower Byte	DMA Controller lower 8-bits of twos complement of the transfer count
3F3F6	DMA TRANSFER COUNT Upper Byte	DMA Controller upper 8-bits of twos complement of the transfer count
3F3F8	DMA MEMORY ADDRESS Lower Byte	DMA Memory Address, low byte
3F3FA	DMA MEMORY ADDRESS Upper Byte	DMA Memory Address, high byte
3F3FC	DMA MEMORY ADDRESS EXTEND	DMA Memory Address, two bit extension of memory address
3F3FE	DMA INTERRUPT VECTOR	Not used in 1720A

Table 2E-2. DMA Controller Control Register

BIT	SYMBOL	FUNCTION
0	RUN	Run/Stop bit. A 1 places the DMA Controller in the run mode. A 0 terminates DMA Controller operation.
1	DIE	Device Interrupt Enable. A 1 allows a high input on DINTR to set the INTR- output low.
2	TOIE	Time-Out Interrupt Enable. A 1 allows the time-out one-shot to set the INTR- output low. The DMA Controller sets the time-out interrupt during a DMA transfer if REPLY- does not go low within 5 us of MSYNC- going low, as discussed later.
3	TCIE	Transfer Count zero Interrupt Enable. A 1 allows a zero in the transfer count register to set the INTR- output low.
4	IOM	Input or Output Mode. A 1 sets READ mode (which sends data from DMA-Floppy Module to Memory). A 0 sets WRITE mode (which sends data from Memory to DMA-Floppy Module). This bit appears on the R/W- pin of the I/C. The BUSWR- output of the Module is RW- inverted.
5	HBUS	Hold BUS. A 1 informs the DMA Controller to hold onto the bus for the entire block instead of releasing the bus after each byte or word transfer. Not used in the 1720A.
6	AECE	Address Extension Carry Enable. A 1 allows a carry from DMA address bit 15 to propogate into bit 16.
7	N/A	Not used.

NOTE

The DMA Controller gates the DMA address onto the DAL (Data Access Lines) outputs in two 9-bit bytes. The first byte out contains MAR bits 8-15 on DAL 0-7 and MAR bit 17 on AE8 (Address Extension 8). The second byte out contains MAR bits 0-7 on the DAL 0-7 lines and MAR bit 16 on AE8. The first byte is valid on the trailing edge of the BUSR- (BUS Request) output and the second byte is valid on the trailing edge of the LAL- (Load Address Low) output.

2E-10. The Status Register (SR) is an 8-bit register that indicates the status of the DMA Controller. Table 2E-3 presents details.

2E-11. Module-To-Memory DMA Transfers

2E-12. This operation takes place when the 1720A reads a block of data from the Floppy Disk into the Memory Module. This topic does not deal with the operation of the Floppy Disk Controller IC or the interface to the Floppy Disk Drive unit which are discussed later. For now, the only concern is the DMA data transfer itself. The following paragraphs cover the operational flow for this transfer.

2E-13. It is assumed that the CPU Module has set up the transfer by: programming the Floppy Disk Controller, loading MAR (Memory Address Register), loading TCR (Transfer Count Register), and placing the appropriate command into the CR (Control Register).

2E-14. When the DMA Controller is in the RUN mode (CR bit 0=1) it waits for a Data Request (DRQ) input) from the Floppy Disk Controller. This event occurs when the Floppy Disk Controller has a data byte from the Floppy Disk ready to go into memory. When DRQ becomes active (high), the BUSR- (BUS Request) output of the DMA Controller becomes active (low). This low acts as an input to the DMA Interrupt Logic block, which responds by asserting the RQOUT- (low) output of the Module to request control of the system buses.

NOTE

If the RQIN- input to the Module is active (low) when DRQ goes active, the DMA Controller waits until RQIN- goes high before bringing BUSR- low.

2E-15. As soon as GRIN- goes low, the DMA Controller has control of the system buses and goes on with the DMA transfer. It controls the direction of the transfer by the status of the R/W- output pin which is tied directly to CR bit-4 (refer to Table 2E-2). The DMA Logic block inverts the R/W output and places the result onto the BUSWR- line of the 1720A Control Bus. Since R/W- is high (Read), BURWR- goes low for this operation to indicate the DMA transfer is sending data to the Memory Module.

Table 2E-3. Status Register

BIT	SYMBOL	FUNCTION
0	BOW	Byte Or Word data channel. A read only bit that indicates the status of the BOW input pin (always high in the 1720A for byte mode).
1	DINT	Device Interrupt. If set to a 1, a device interrupt has occurred (set by the Floppy Disk Controller IC to indicate command complete). This is a read/write bit. Resetting this bit to a zero resets the INTR- (Interrupt Request) output of the DMA Controller. Refer to DMA Interrupt topic.
2	TOI	Time-Out Interrupt. If set to a 1 then a time out error has occurred. This is a read/write bit. Resetting this bit to a zero resets INTR-. Refer to DMA Interrupt topic.
3	TCZI	Transfer Count = Zero Interrupt. If set to a 1 then a transfer count equal zero interrupt has occurred. It is a read only bit that sets the EOB (End of Block) output when set. Refer to the DMA Interrupt topic.
4	IOM	Input-Output Mode. A read only bit that reflects the status of bit 4 in the Control Register.
5	HBUS	Hold Bus. A read only bit that reflects status of bit 5 in the Control Register. Not used in 1720A operation.
6	AECE	Address Extension Carry Enable. A read only bit that reflects the status of bit 6 in the Control Register.
7	BUSY	Busy (data transfer not complete). A read only bit that reflects the status of bit 0 in the Control Register.

Note: Bit 1, 2, 3, are set if the corresponding condition occurs. The enable bits in the CR to affect only the INTR- output and not the Status Register.

2E-16. The DMA Controller places the high byte of the memory address on the DAL lines, then sets BUSR- to 1. This causes two things to happen:

1. Address Buffer U48 latches the DAL address.

2. U41 in the DMA Logic block latches the state of the AE8 output. The output of U41 becomes AD17 on the 1720A Address Bus.

2E-17. The DMA Controller places the low byte of the Memory address on the Dal lines, then sends the LAL- (Load Address Low) and MSYNC- (Memory Sync) low pulses. This causes two things to happen:

1. The trailing edge of LAL- clocks the Address Buffer U47. This latches the DAL address. In addition, the trailing edge of the pulse clocks U41 to latch the AE8 output (for the AD16 line of the 1720A Address Bus).

2. MSYNC-goes low and places a low on the BMODE- 1720A Control Bus line. Also, this low sets AD16, AD17, ADVAL- and BUSWR- on the 1720A Buses into the states determined by the DMA Controller. The state of ADVAL- is the same as the state of MSYNC- (low) but lags MSYNC- by a delay determined by the Timing Logic. This places the address on the Address Bus slightly ahead of the ADVAL- active state.

2E-18. The DMA Controller then places the DAL lines in a high impedence state in anticipation of a data transfer to the 1720A Bus from the Floppy Disk Controller.

2E-19. The DMA Controller activates RE- (Read Enable). This low level to the Floppy Disk Controller enables data flow from that device to the Data Buffers. At this time, the 1720A Control Bus Signal BUSWR- is low during the time MSYNC- is low and is an input to U27 pin 6. As a result, BUSWR- enables one input to NAND gate U32 pin 6 in the Data Buffer Block. The MSYNC- low enables the other leg and the output (pin 4) acts through U31 pin 4 to place the Data Buffers into the output mode.

2E-20. At this time several events take place at the same time:

1. The complete 18-bit address is present on the 1720A Address Bus.

2. ADVAL- on the 1720A Control Bus is low to inform the Memory Module of this fact.

3. BUSWR- on the 1720A Control Bus is low to indicate that data flow is from the DMA to Memory (write operation).

4. An 8-bit data byte is present on the 1720A Data Bus.

2E-21. The DMA Controller waits for ADACK- (Address Acknowledge) from the Memory Module to make its REPLY- input active. When this happens the DMA Controller deactivates its RE- output. See the Time-Out portion of the DMA Interrupts topic.

2E-22. MSYNC- now goes high and the DMA-Floppy Module gives up control of the bus.

2E-23. After the completion of every data transfer, the DMA Controller increments the Memory Address Register by one, in preparation for the next transfer.

2E-24. After the completion of every data transfer, the DMA Controller increments the Transfer Count Register by one. The DMA Controller considers transfers to be completed when this count equals zero.

2E-25. Memory-to-Module DMA Transfers

2E-26. This operation takes place when the 1720A writes a block of data onto the Floppy Disk from the Memory Module. Again, this topic does not deal with the operation of the Floppy Disk Controller or the interface to the Floppy Disk Drive unit which are discussed later. For now, the only concern is the DMA data transfer itself.

2E-27. The operational flow of this operation is nearly the same as the Module-to-Memory flow just described. The paragraphs that follow cover the operational flow.

2E-28. It is assumed that the CPU Module has set-up the transfer by: programming the Floppy Disk Controller, loading MAR (Memory Address Register), loading TCR (Transfer Count Register), and placing the appropriate command into the CR (Control Register).

2E-29. When the DMA Controller is in the RUN mode (CR bit=1) it waits for a Data Request (DRQ) input from the Floppy Disk Controller. This event occurs when the Floppy Disk Controller is ready for a data byte from the Memory Module which would go onto the Floppy Disk. When DRQ becomes active (high), the BUSR- (BUS Request) output of the DMA Controller becomes active (low). This low acts as an input to the DMA Interrupt Logic block, which responds by asserting the RQOUT- (low) output of the Module to request control of the system buses.

NOTE

If the RQIN- input to the Module is active (low) when DRQ goes active, the DMA Controller waits until RQIN- goes high before bringing BUSR- low.

2E-30. As soon as GRIN- goes low, the DMA Controller has control of the system buses and goes on with the DMA transfer. It controls the direction of the transfer by the status of the R/W- output pin which is tied directly to CR bit 4 (refer to Table 2E-2). The DMA Logic block inverts the R/W- output and places the result onto the BUSWR - line of the 1720A Control Bus. Since R/W- is low (write), BUSWR- goes high for this operation to indicate the DMA transfer is receiving data from the Memory Module.

2E-31. The DMA Controller places the high byte of memory address on the the DAL lines, then sets BUSR- to 1. This causes two things to happen:

1. Address Buffer U48 latches the DAL address.

2. U41 in the DMA Logic block latches the state of the AE8 output. The output of U41 becomes AD17 on the 1720A Address Bus.

2E-32. The DMA Controller places the low byte of the Memory address on the DAL lines, then sends the LAL- (Load Address Low) and MSYNC-(Memory Sync) low pulses. This causes two things to happen:

1. The trailing edge of LAL- clocks the Address Buffer U47. This latches the DAL address. In addition, the trailing edge of the pulse clocks U41 to latch the AE8 output (for the AD16 line of the 1720A Address Bus).

2. MSYNC- goes low and places a low on the BMODE- 1720A Control Bus Line. Also, this low sets AD16, AD17, ADVAL- and BUSWR- on the 1720A Buses into the states determined by the DMA Controller. The state of ADVAL - is the same as the state of MSYNC- (low) but lags MSYNC- by a delay determined by the Timing Logic. This places the address on the Address Bus slightly ahead of the ADVAL- active state.

2E-33. The DMA Controller then places the DAL lines in a high impedence state in anticipation of a data transfer to the 1720A Bus from the Floppy Disk Controller.

2E-34. The DMA Controller activates WE - (Write Enable). This low level to the Floppy Disk Controller enables data flow from the Data Buffers into the Floppy Disk Controller. At this time, the 1720A Control Bus Signal BUSWR is high during the time MSYNC- is low and is an output of U27 pin 14. This high level becomes a low level at U33 pin 4. As a result, BUSWR- (high) enables one input to NAND gate U32 pin 11. The MSYNC- low enables the other leg and the output (pin 13) acts through U31 pin 1 to place the Data Buffers into the input mode.

2E-35. At this time several events take place at the same time:

1. The complete 18-bit address is present on the 1720A Address Bus.

2. ADVAL- on the 1720A Control Bus is low to inform the Memory Module of this fact.

3. BUSWR- on the 1720A Control Bus is high to indicate that data flow is from the DMA to Memory (write operation).

4. An 8-bit data byte is present on the 1720A Data Bus.

2E-36. The DMA Controller waits for ADACK- (Address Acknowledge) from the Memory Module to make its REPLY- input active. When this happens the DMA Controller de-activates its RE- output. See the Time-Out portion of the DMA Interrupts topic.

2E-37. MSYNC- now goes high and the DMA- Floppy Module gives up control of the bus.

2E-38. After the completion of every data transfer, the DMA Controller increments the Memory Address Register by one, in preparation for the next transfer. It also increments the Transfer Count Register by one. The DMA Controller considers transfers to be completed when this count equals zero.

2E-39. DMA Interrupts

2E-40. There are three individually enabled interrupt conditions associated with the DMA Controller. If any of the conditions occur, it sets the corresponding bit in the Status Register. If the appropriate enable bit in the Command Register is set, then the INTR - output of the Controller also goes active. If this happens, the interrupt Logic block produces a low on DI02 during the next RINT cycle to give an interrupt to the CPU Module.

2E-41. Once an interrupt condition sets its corresponding bit in the status register, the bit stays set until a CPU write to the Status Register occurs with a zero in the bit position. For a Transfer-Count-Equals-Zero to be cleared, the Transfer Count Register must be loaded with a non-zero count.

2E-42. A detailed description of the three interrupt conditions follows:

1. A Device Interrupt (DINT) condition occurs when the DINTR input to the DMA Controller goes high. This sets SR bit 1 and, if CR bit one is set, it activates INTR-. In addition, the Controller resets the RUN bit to zero to terminate all

subsequent DMA transfers. The interrupt service routine in the CPU Module clears the DINT status bit.

2. Transfer Count Equals Zero (TCZI). When the DMA Controller increments the TCR to zero after a DMA transfer, it sets the TCZI status bit (SR bit 3) and resets the RUN bit (CR bit 0) to terminate all DMA operations. If CR bit-3 is set, it activates INTR-.

3. Time-Out Interrupt (TOI). During any DMA transfer, the leading edge of MSYNC- triggers an internal time delay of approximately 5 microseconds. If the DMA Controller does not receive an active low REPLY (ADACK - from the Control Bus) within that time delay, it terminates the DMA operation, resets the RUN bit, and sets the TOI status bit (SR bit-2). If CR bit-2 is set, it activates INTR -.

2E-43. FLOPPY DISK CONTROLLER GENERAL DESCRIPTION

2E-44. As shown on Figure 2E-1, the Floppy Disk Controller interfaces the Floppy Disk Drive to the DMA Controller and to the 1720A Buses. The discussion that follows first considers the interface to the DMA Controller and the 1720A buses, then considers the interface to the Floppy Disk Drive unit. Following this, the discussion puts everything together by going through both read and write disk operations. Refer to both the Schematic and to the DMA Floppy Disk Interface Block Diagram as the discussion progresses.

2E-45. DMA and System Interface

2E-46. The interface to the DMA Controller and to the System Buses is accomplished through the eight DAL (Data Access Lines) and associated control signals. The Floppy Disk Controller IC (called FC from now on) uses the DAL- lines to transfer Data, Status, and Control words. The DAL- buffers are tri-state buffers that function as output drivers when the CS- (Chip Select) and the RE- (Read Enable) inputs are low andfunction as input receivers when CS- and WE- (Write Enable) inputs are low.

2E-47. As shown in Table 2E-1, there are four registers in the FC that are accessed via Memory maped I/O operations by the CPU Module:

1. Status Register (read)/Command Register (write)

2. Track Register (write and read)

3. Sector Register (write and read)

4. Data Register (write and read)

2E-48. Before each DMA operation, the CPU Module loads these registers as necessary to complement the command it places into the command register. The read or write reference in parenthesis after the register name in the previous list indicates what the CPU Module does to that register. The commands are discussed in the paragraphs under the Commands heading. During the DMA operation, the Data Register serves the functions already covered under the DMA portion of this discussion.

2E-49. Commands

2E-50. The FC accepts 11 commands. Command words must be loaded into the Command Register when the Busy status bit (bit 0) is set to 0. The one exception is the Force Interrupt command (discussed later). Throughout the execution of a command, the Busy status bit is equal to 1. At the completion of a command, the FC sets the Busy status bit to a 0 and issues INTRQ. The Status Register indicates whether the completed command encountered an error or was error free. For ease of discussion, this topic talks about these commands as if there were four types. Table 2E-4 presents the four types of commands and their binary code. Table 2E-5 presents the meanings of the various flag bits in the commands. The paragraphs that follow offer an explanation of the 11 commands.

2E-51. The Restore (seek track 00) command causes the FC to move the read/write head over track 00 and set the Track Register to all zeros. It issues 255 stepping (maximum) pulses to the Disk Drive and, if it does not find track 00 with this effort, sets the Seek Error status bit, then terminates the operation and activates INTRQ.

Table 2E-4. Command Summary

TYPE	COMMAND	BITS
		7 6 5 4 3 2 1 0
I	Restore	0 0 0 0 h V r1 r0
I	Seek	0 0 0 1 h V r1 r0
I	Step	0 0 1 u h V r1 r0
I	Step In	0 1 0 u h V r1 r0
I	Step Out	0 1 1 u h V r1 r0
II	Read Sector	1 0 0 m X E 0 0
II	Write Sector	1 0 1 m X E X a0
II	Read Address	1 1 0 0 0 1 0 0
III	Read Track	1 1 1 0 0 1 0 X
III	Write Track	1 1 1 1 0 1 0 0
IV	Force Interrupt	1 1 0 1 I3 I2 I1 I0

X = Don't care Note: Bits shown in TRUE form.

Table 2E-5. FLAG Summary

TYPE I
h = Head Load Flag (Bit 3) h = 1, Load head at beginning h = 0, Unload head at beginning V = Verify flag (Bit 2) V = 1, Verify on last track V = 0, No verify r1r0 = Stepping motor rate (Bits 1-0) u = Update flag (Bit 4) u = 1, Update Track register u = 0, No update
TYPE II
m = Multiple Record flag (Bit 4) m = 0, Single Record m =1, Multiple Records a0 = Data Address Mark (Bit 0) a0 = 0, FB (Data Mark) a1 = F8 (Deleted Data Mark) E = 15 ms Delay E = 1, 15 ms delay E = 0, no 15 ms delay
TYPE IV
Ii = interrupt Condition flags (Bits 3-0) I0 = 1, Not-Ready to Ready Transition I1 = 1, Ready to Not-Ready Transition I2 = 1, Index Pulse I3 = 1, Immediate Interrupt

NOTE

If the 1720A Control Bus signal DCOK- goes active (low), the FC executes this command when DCOK- goes inactive (high).

2E-52. The Seek command assumes that the Track Register contains the correct track number of the current position of the read/write head and the Data Register contains the desired track number. The FC updates the Track Register while issuing stepping pulses in the appropriate direction until the contents of the Track Register are equal to the contents of theData Register.

2E-53. The Step command causes the FC to issue one stepping pulse to the Disk Drive. The stepping motor direction is determined by the last step command (Step In or Step Out) given by the DMA Floppy Module. The Step In command causes the FC to issue one stepping pulse in the direction towards track 76. The Step Out command causes the FC to issue one stepping pulse in the direction of track 00.

2E-54. The Read Sector command assumes that the CPU Module has loaded the Sector Register with the desired sector number. The FC sets the Busy Status Bit and when an ID field is encountered that has the correct track number, correct sector number, and correct CRC, (Cyclic Redundency Check) the FC presents the data field to the CPU Module via the DMA process. It must find the data address mark of the data field within 43 bytes of the last ID field CRC byte; if not, the FC sets the Record Not Found status bit (bit 3) and terminates the operation.

2E-55. The Write Sector command assumes that the CPU Module has loaded the Sector Register with the desired sector number. The FC sets the Busy status bit and when an ID field is encountered that has the correct track number, correct sector number, and correct CRC, (Cyclic Redundency Check) it sets DRQ. The FC counts off 22 bytes from the CRC field, activates the WG (Write Gate) output, then writes 12 bytes of zeros on the disk. At this time it writes the Data Address Mark on the disk as determined by bit 0 in the command. See Table 2E-5. The FC then writes the data field, generating the necessary DRQs to the DMA Controller. After the FC writes the last data byte on the disk, it writes the two CRC bytes. Then it deactivates the WG signal, and sets INTRQ.

2E-56. The Read Address command causes the FC to load the read/write head and set the Busy status bit. Then the FC reads the next ID field and assembles the next six data bytes in this field. During assembly it transfers them one at a time to the Data Register, and generates a DRQ for each byte for each transfer. The six bytes contain:

Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6
Track Addr.	Side Number	Sector Addr.	Sector Length	CRC	CRC

2E-57. The Read Track command causes the FC to load the head and set the Busy Status bit. Reading starts with the leading edge of the first encountered index mark and continues until the next index pulse. As each byte is assembled, it is transferred to the Data Register, and Data Request is generated for each byte transferred. The FC does not perform a CRC and includes gaps in the input data stream. It accumulates bytes in sychronization with each Address Mark encountered. Upon completion of the command, the FC activates the interrupt.

2E-58. The Write Track causes the FCL to set the Busy status bit, load the read/write head and start writing with the leading edge of the first encountered index pulse and continue until the next index pulse. At the completion of this, it activates INTRQ. Even though the FC sets DRQ immediately upon receiving this command, it does not start writing until after the first byte is in the Data Register. If this does not happen by the time it encounters the index pulse, the FC terminates the operation, resets the Busy status bit, and sets the Lost Data Status bit. During the operation, if a data byte is not in the DR when needed, the FC substitutes a byte of zeros. The FC writes the CRC bytes and Address marks upon receipt of certain special characters in the incoming data stream.

2E-59. The Force Interrupt is different from the others. The CPU Module can load this command into the Command Register at any time. If there is a current command under execution, the FC terminates that command and generates an interrupt when it detects the condition specified in the I0 through I3 field, as follows:

I0 = Not-Ready-To-Ready Transistion
I1 = Ready-To-Not-Ready Transition
I2 = Every Index Pulse
I3 = Immediate Interrupt

NOTE

If I0 - I3 = 0, there is no interrupt generated, but the FC terminates the current command and resets the Busy status bit. This is the only command that will clear the immediate interrupt.

2E-60. Floppy Disk Interface

2E-61. Refer to Figure 2E-1. The FC provides drive control directly to the Floppy drive unit, but uses the Write Precompensation block to provide write data and uses the Clock Recovery block to recover the read data. In addition to the FC, there is a Medium Status Logic block, a Drive Select block, and a Motor Shut Off block. This discussion covers each of these blocks as preparation for the Read and Write Operation discussions that follow. Refer to the Schematic when the discussion mentions specific signals, components and pin numbers.

2E-62. The FC has two modes of operation according to the state of pin 37. When this pin is a 1 (high), single density is selected and when it is a 0 (low), as in the 1720A, double density is selected. In either case the CLK (Clock) input frequency is 2 MHz. However, when interfacing with a small floppy (as in the 1720A), the clock input frequency is set to 1 MHz for both single density and double density. This gives stepping rates of 6, 12, 20, and 30 ms.

2E-63. Four commands (from the CPU Module) cause the FC to position the Read-Write head in the Floppy Drive (see Commands). The period of each positioning step cycle is specified by the r field in bits 1 and 0 of the command word (see ro r1 in Table 2E-4).

2E-64. The Step output (STEP) of the FC is a 4 us pulse output that becomes ISTP at the drive connector P9. For every step pulse issued, the drive moves one track location in a direction determined by the Direction output. The Direction output (DIRC) of the FC is high when stepping in and low when stepping out. It is valid 24 us before the first stepping pulse. It becomes IDIR at the drive connector P9.

2E-65. The WD (Write Data) output signal from the FC goes to the Write Precompensation block. The write data output of this block becomes the IWDA output at the drive connector P9. The Write Precompensation block determines the actual value of the write precompensation necessary for correct read back of the data bits.

2E-66. Both the RWR (Raw Read Data) and the RCLK (Read Clock) inputs to the FC come from the Clock Recovery block. This block receives read data (IRDA) from the Floppy Disk Drive in a MFM (Modified Frequency Modulation) format. It functions as a data separator and provides the RWR and RCLK inputs for the FCF.

2E-67. The Drive Select block responds to a CPU Module memory maped I/O write into the Drive Unit Select/Medium Status Register (address 3F3E8), bits 0, 1, 2, 3. This selects which of the Drive Units 0, 1, 2, 3, that the CPU Module desires to use during the next read or write operation. The Drive Select block also activates the HLT (Head Load Timing) input to the FC to allow time for completion of the head loading process in the Floppy Disk Drive Unit. This process occurs each time the CPU Module selects a new drive.

2E-68. As shown in Figure 2E-1, the disk inserted switch input (P9 pin36) to the Module from the Floppy Disk Drive Unit goes to two blocks: the Motor Shut Off block and the Medium Status Logic. The functions of these two blocks are:

1. The Motor Shut Off Block responds to any memory mapped I/0 access of the DMA-Floppy I/F Clock Module by:

   a. Activating the ISLT output of P9 that goes to the selected drive if the switch input from the Drive indicates there is media present. The Drive responds by starting its motor and loading its read/write head against the media.

b. Activating the RDY (Ready) input to the FC after a delay to allow the motor to come up to speed. The FC samples this input and does not initiate any read or write operation until RDY goes high.

c. Counting the incoming index pulses (IINXP) which come from the Drive at approximately 200 ms intervals and shutting the motor off if the count reaches 80 (16 seconds).

d. Shutting off the motor if the switch input from the Drive opens (media removed).

2. The Medium Status Logic sets bit 0 in the Drive Unit Select/Medium Status Register if a disk is in the Drive and sets bit 1 each time the disk is changed.

2E-69. Head Positioning

2E-70. As mentioned previously, there are four commands that cause head positioning (see Commands). The CPU Module positions the read/write head over the correct track with appropriate commands into the command register of the FC which generates the STEP and DIRC outputs as necessary. The FC uses the input TR00 (ITRK0 at P9) which goes active (low) when the read/write head is over track zero to implement the Restore command. The CPU Module positions the read/write head before initiating read or write operations. The Track Register always has the current track address.

2E-71. Disk Read Operations

2E-72. The 8-bit Sector Register (see Table 2E-1) holds the address of the desired sector position. The FC compares the contents of this register with the recorded sector number in the ID field of the sectors in the track.

2E-73. The RWR input to the FC comes from the Clock Recovery block and is a 250 ns pulse per flux transition. Along with this signal, the FC receives a RCLK signal that indicates flux transition spacings. An 8-bit shift register assembles serial data from the RWR input into an 8-bit parallel byte of data.

2E-74. When the RE- input goes low, the FC transfers the assembled data byte from the Shift Register into the Data Register and from there into thedata buffers on the DAL output lines. As mentioned in the DMA portion of this write up, the FC issues a DRQ during a read to indicate that a complete byte of data is ready. The DMA Controller responds by setting up a write memory DMA operation at the correct address and making the RE- input to the FC low. Any new command resets the INTRQ output.

2E-75. Disk Write Operations

2E-76. Head positioning and sector addressing are the same as in a read. As soon as the FC finds the sector address wanted, it issues the DRQ output to the DMA Controller which responds by initiating a memory read DMA operation at the correct address and placing a byte of data on the DAL lines.

2E-77. The DMA Controller activates the WE- input to the FC, and the FC transfers the byte of data from the DAL lines to the Data Register. From here the data goes into the Data Shift register. The FC then shifts the data out of the Data Shift register to the WD (Write Data) output. As soon as the Data Shift Register is empty, the FC issues another INTRQ to the DMA Controller for the next byte of data.

Section 2F
Power Supply

2F-1. INTRODUCTION

2F-2. The 1720A Power Supply provides all the operating voltages required by the other modules and provides battery derived supply voltages for powering the electronic disk. Figure 2F-1 shows the three PCBs that make up the Power Supply and shows the functional blocks within each PCB.

2F-3. SWITCHING POWER SUPPLY

2F-4. The Switching Power Supply operates at 50 kHz. It operates over a line frequency and voltage range of:

VOLTAGE RANGE	LINE FREQUENCY
90V to 132V	47 Hz to 63 Hz
192V to 250V	47 Hz to 63 Hz
104V to 126V	380 Hz to 420 Hz

2F-5. It provides the following output voltages:

5VDC at 8A
12VDC at 1.5A
12VDC at 2A
-12VDC at 1A
70VDC at 0.4A

2F-6. The components that make up the Switching Power Supply span all three PCBs. The paragraphs that follow present the theory of operation for this supply. Refer to both the Block Diagram and to the schematic as the discussion progresses. When referring to the schematic, notice that the particular PCB (or PCBs) on any given page of the schematic is indicated by a four digit number. These numbers mean:

1006 Power Supply Control PCB
1007 Power Supply Rectifier PCB
1016 Switching Transistor PCB

2F-7. ±HV SUPPLY

2F-8. The ±HV Supply is on the P/S Control PCB. The supply consists of the Line Transient Suppression Block, the Inrush Current Limit Block, the Rectifier and Filter, and the Common Mode Filter.

2F-9. The incoming line voltage enters the supply through F1 and passes through the Line Transient Suppression circuit. Relay K1 is open at this time (start up), therefore, the current passes through RT1 which is a thermistor offering a higher resistance while cold. This provides the Inrush Current Limit function. Diode CR4 rectifies the line voltage and, if S3A is in the 240V position, acts as a bridge rectifier. If S3A is in the 120V position, CR4 acts as a voltage doubler. In either case, the supply capacitors C1 and C2 charge up to ±170V. The Common Mode Filter provides appropriate filtering for the final ±170V output on P18 pins 1 (+HV) and 5 (-HV).

2F-10. Notice that the relay K1 closes as the -12V supply comes up. The only function of K1 is to bypasss RT1 after the supply is up and regulating. This allows RT1 to cool off and be ready to provide inrush current protection again if the supply shuts down and comes back up. In addition, the bypass eliminates the power loss caused by the incoming current passing through RT1.

2F-11. The Spark Gap E1 is used only in the 120V mode. Its function is to breakdown and blow the fuse if the supply is plugged into 240V while S3 is in the 120V position. This prevents damage to the supply components.

2F-12. The ±HV Outputs just discussed go directly to the Switching Transistor PCB. Before going into the theory of operation for this PCB and the power Supply Rectifier PCB, it is necessary to discuss the Pulse Width Modulator that provides the input to the Drive Circuit.

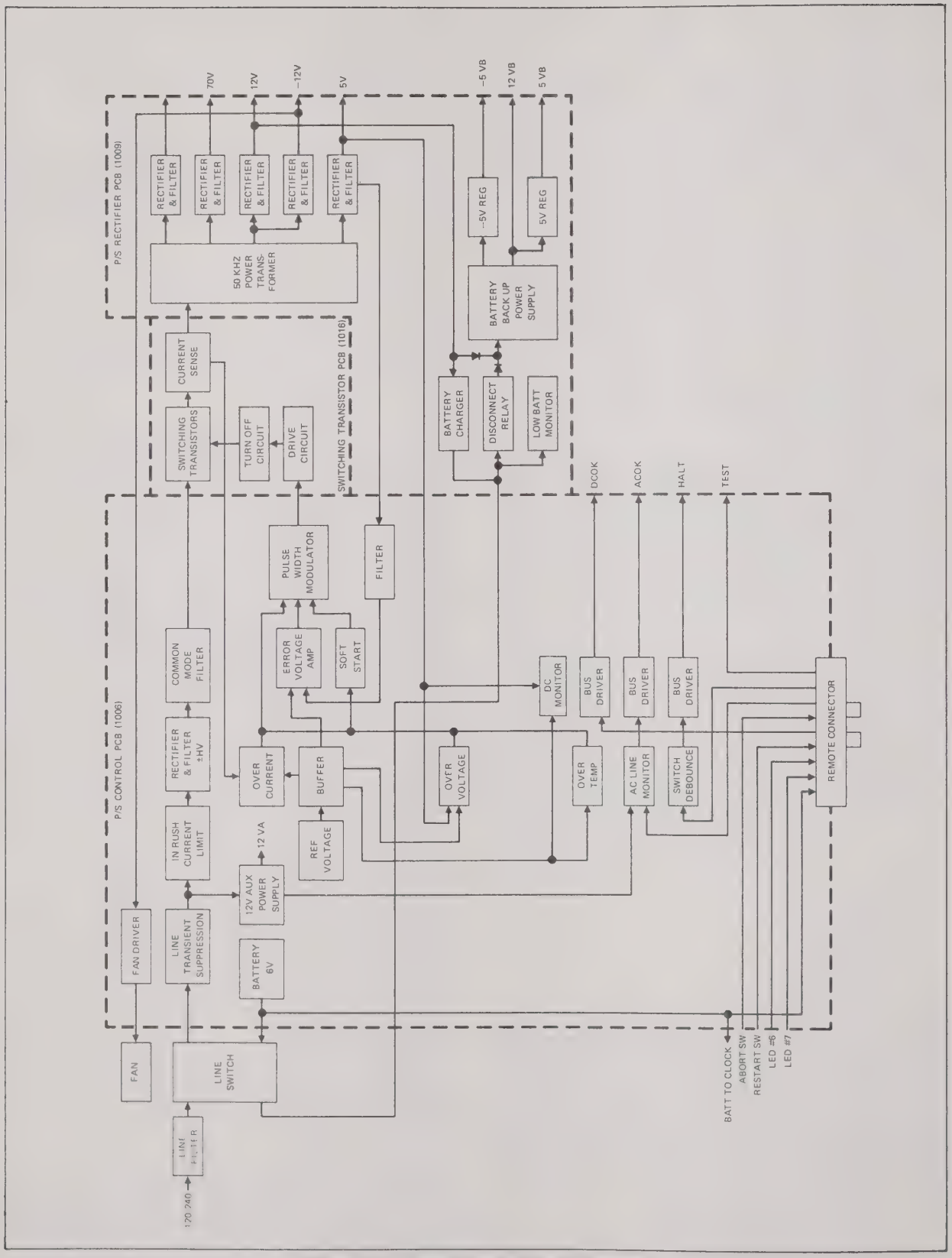


Figure 2F-1. Power Supply Block Diagram

2F-13. PULSE WIDTH MODULATOR AND CONTROL

2F-14. As soon as S1 closes, incoming power goes to the Start Up Supply as well as the ±HV Supply. The Start Up Supply, consisting of T2, CR1, U4 comes up must faster than the ±HV Supply. It provides +12V through CR13 to activate the Pulse Width Modulator and Control circuits during the time it takes the main +12V supply to come up. As soon as the main +12V supply voltage is up to +12V, CR13 is reverse biased effectively disconnecting the Start Up Supply.

2F-15. The Pulse Width Modulator IC, U5, responds to the sense input on pin 6 by changing the width of the 50 KHz output pulses on pins 11 and 13.As the voltage on pin 6 goes down, the pulse width lengthens. This provides regulation of the 5V Supply (discussed in greater detail later). The two outputs of U5 are always equal widths but are 180 degrees out of phase with each other.

2F-16. During the time it takes C13 to charge up to 12V, the current through R18 provides a sense input to U5. This input causes the pulse width out of U5 to begin at a narrow width and slowly move outward to provide the Soft Start function.

2F-17. R22 must be adjusted to establish the 50 kHz fundamental frequency of the output at U5 pins 11 and 13. This is necessary for the other components in the supply to function correctly.

2F-18. Switching Transistor PCB

2F-19. The output of the ±HV Supply and the 50 kHz output of the Pulse Width Modulator both act as inputs to the Switching Transistor PCB. The +HV enters the PCB on J18 pin 1 and goes to the collector of one of the switching transistors, Q1. The -HV enters on pin 5 and goes to the emitter of the other switching transistor Q2.

2F-20. The complementary 50 kHz outputs of the Pulse Width Modulator enter the PCB on J18 pins 13 and 15 to provide the drive for Q5 and Q6. As these transistors turn off and on, their outputs work through coupling transformers T2 and T3 to turn Q1 and Q2 on and off (180 out of phase with each other) for the duration of the pulses provided by the Pulse Width Modulator.

2F-21. The transistors Q1 and Q2 are connected together and provide the active drive to one end of the primary winding of the power transformer on the PS Rectifier PCB. The other end of the power transformer primary goes to the power supply neutral through the single turn primary of the Current Sense transformer T1. Capacitor C2 prevents any DC component in this current path.

2F-22. Transistors Q3 and Q4 provide the Turn Off Circuit function by actively depleting the emitter-base junction at turn off time. If these transistors are not operating, a significant increase in the turn off fall time of the outputs of Q1 and Q2 occurs which causes the power being dissipated by these transistors to rise. This decreases the overall efficiency of the supply and shortens the life span of Q1 and Q2.

2F-23. The output of the Current Sense transformer is a positive and negative going train of pulses at 50 kHz. These pulses have an amplitude that is proportional to the current in the primary of the Power Transformer, a width equal to the width of the output pulses of the Pulse Width Modulator and a polarity equal to the direction of current in the primary. The output of Q1 produces the negative going pulse and the output of Q2 produces the positive pulse. This wave form can be monitored over on the Power Supply Control PCB at TP1.

2F-24. 50 kHz Power Transformer

2F-25. Refer to the P/S Rectifier Schematic. The 50 kHz Power Transformer has 4 secondaries. Its small physical size is due to the high frequency it is designed to run at. A transformer of similar power capabilities designed for 60 Hz would be many times larger.

2F-26. Supply Rectifiers

2F-27. Notice how each of the output voltages has its own individual rectifier and filter. Since all supplies originate from secondary windings that share the same transformer core and primary winding, they all respond to pulse width modulation of the primary equally. Because the Pulse Width Modulator senses and regulates the 5V supply output (discussed in detail later), these outputs are all slaved to the 5V supply output.

2F-28. +5V Sense and Regulation

2F-29. Notice how the 5V Sense High/Low outputs originate ahead of the power supply filter. Now, refer to the Power Supply Control schematic and note how these same signals (5V SENSE and 5V RETURN) act as inputs to a filter circuit which reacts to load changes faster than sensing at the load itself. The output of this circuit acts as an input to the non-inverting input of the Error Amp U3. The other input to U3 is a reference voltage that can be adjusted by R50 (5 VOLT ADJ). Since U3 acts like a comparator, its output will swing up and down as the non-inverting input goes above and below the reference voltage.

2F-30. CR11 diode couples the output of U3 to the input of the Pulse Width Modulator U5 (pin 6). As the output of U3 changes, U5 regulates the pulse width of its

drive signal outputs on a pulse to pulse basis to keep the 5V supply output in regulation. Since line regulation is provided with respect to the 5V supply, all other output voltages track the 5V output.

2F-31. Because there is no remote sensing, load regulation is dependent on distribution impedances. The power supply must not be operated without a minimum load of about 10% on all sources. When it is unplugged from the controller, an interlock prevents operation.

2F-32. FAULT PROTECTION

2F-33. The power supply will shut down in response to three different fault conditions; primary circuit overcurrent, +5V over voltage, and high temperature. Detection of one (or more) of these faults will trigger a monostable circuit consisting of U1, R56-58, CR7,CR8, C23 and C38. The output of this circuit remains active for about one second. As long as it is active, the modulator output will be held off via the soft-start circuit. If the cause of the fault no longer exists when the monostable active period is complete (i.e., the modulator inhibit input is not active low),the power supply will attempt to restart automatically invoking a soft start.

2F-34. LOW LINE SHUTDOWN

2F-35. In order to avoid excessive ripple on the supplies (incurred by dropping out of regulation at low line voltages), the power supply is held off (shut down) until the line voltage is sufficiently high. The AC line sense filter is divided by R29, R30, and C15 and used as an input to a comparator (U2 pin 10). Shutdown occurs at a lower line voltage than turn-on to avoid oscillation.

2F-36. OVERCURRENT PROTECTION

2F-37. Overcurrent protection limits the total power dissipation of the switching transistors. Refer to the Power Supply Control schematic. Connector P18 pin-12 is the output of the current sense transformer on the Switching Transistor PCB. This putput establishes a voltage level on the integrator C16/R46 that is proportional to the current being drawn by the Power Transformer. Both the threshold voltage and the voltage on C16 are compared by the Over Current comparator U1. If the threshold is exceeded, the low output of U1 causes a shutdown by bringing pin-15 of U5 low.

2F-38. OVERVOLTAGE PROTECTION

2F-39. U1 pin-10 input is proportional to the 5V supply voltage, from P14 pin-27. If the 5V supply voltage exceeds the threshold established by R9 and R10, the output of U1 pin-13 goes low and causes U5 to shut down the supply by bringing U5 pin-15 low.

2F-40. FAN

2F-41. The fan electronics are in a separate module that has the -12V supply voltage as an input. This module provides three phase drive power for the fan that makes the fan speed independent of line frequency. This is done to allow the supply to run on a variety of line frequencies.

2F-42. BATTERY BACKUP

2F-43. Battery Backup Supply

2F-44. As shown on Figure 2F-1, the Battery Backup Power Supply receives input power from the 6V Battery and the 12V Supply. Each of these power sources feeds the Battery Backup Power Supply through a diode. As long as the 12V supply is up and running it reverse biases the diode for the 6V Battery and acts as the power source for the supply. If it drops below 6V (nominal) the Battery Backup Supply draws power from the Battery.

NOTE

The 6V Battery has a fuse which protects it from damage when an external battery of greater than 6V is connected. If this fuse (F1, 5A) blows, the Battery Backup Supply will not work off the internal Battery until it is replaced.

2F-45. Refer to the P/S Rectifier Schematic. The input for the supply is through R6. Notice how both sources meet here and how power from either one can flow through R6, through the primary of T4, and through Q3 to ground when Q3 is on. IC U2 functions as a switching regulator for the +12VB supply by turning Q3 on and off as follows:

1. Since an internal oscillator controls the basic rate of operation for U2, we can begin this discussion with U2 providing a current path through R7 to the base Q3 which turns Q3 on. As soon as Q3 is on, current begins to ramp upwards through R6 and the primary of T4. At approximately 8A the voltage drop across R6 exceeds the current sense point of U2, across pins 13 and 14.

2. At that time, U2 opens the bias path for the emitter base current that turned Q3 on. As soon as Q3 turns off, the primary of T4, acting as an inductor, causes the voltage on the collector of Q3 to rise. This voltage causes a current flow through CR10 which begins to charge C19.

3. Pin 10 of U2 monitors the rising voltage on C19 through a divider consisting of R20, R21 and R23 (the +12VB Adjustment). As soon as this voltage

reaches +12V, U2 turns Q3 on again to repeat the process. Note that the time it takes to charge C19 is load dependent and that as the load increases, the rate at which U2 switches Q3 on and off increases.

2F-46. IC U2 controls Q2 directly causing Q2 to act as a series pass regulator for the +5VB Supply. Pin 7 of U2 is the 5VB sense, and a reference voltage for U2 pin-6 comes from the junction of R5 and R4.

2F-47. The transformer action of T4 provides a negative source voltage for the -5V regulator U4. CR11 provides rectification of the output of T4.

2F-48. Battery Charger

2F-49. IC U3 is a linear voltage regulator which is used in conjunction with Q1 to provide a charging source for the battery when ac power is present. This charging circuit is current limited to about 250 milliamps. Resistor R25 provides adjustment of battery charging voltage.

2F-50. Low Battery Dropout

2F-51. IC U1 monitors the battery voltage and opens K1 when the battery voltage drops below 5.5V. This disables the Battery Backup Supply before the battery is fully discharged so that there is still some capacity to maintain the time of day circuitry. Switch S1 is used to manually disable the battery backup feature when it is not necessary. Note that if this circuit is always enabled, the battery will discharge each time ac power is shut off. This will reduce battery life. It also reduces the capacity of the battery to backup the E-disk.

2F-52. STATUS LINES

2F-53. Three bus lines can be set by the power supply:

1. ACOK set false (low) when the line power is lost for more than 30 ms. Also set false if the ABORT switch is enabled and closed.

2. DCOK set true (high) about 1/2 second after the power switch is turned on and set false (low) with any power shutdown command.

3. HALT controlled through a debounce circuit by connecting a switch to a pin on the remote connector on the rear panel.

2F-54. REMOTE CONNECTOR

2F-55. The remote connector provides the following features:

1. ABORT switch enable and remote operation

2. HALT control

3. RESTART switch enable and remote operation

4. External Battery

5. LED #6 and LED #7 which can be set under software control.

6. Remote operation of battery backup disable

Section 2G
Clock Module

2G-1. INTRODUCTION

2G-2. The 1720A Clock Module resides on the DMA Floppy Interface Module PCB and does not occupy a separate slot in the mother board. The CPU Module uses the 1720A CRU Bus to communicate with the Clock Module. The clock is provided with battery back up from a 6 volt battery during power off and will operate normally for about 2 to 3 months. It will keep track of time through the year 2004 but does not automatically take leap years into account and must be manually reset each leap year.

2G-3. CRU INTERFACE

2G-4. As mentioned in the Introduction, the CPU Module uses the CRU Bus to communicate with the Clock Module. This usage of the CRU Bus involves 15 bits, each with a specific CRU address. The CPU Module can write (set/reset) or read each bit. All operations involving the Clock Module are carried out via these 15 bits. Refer to the DMA Floppy I/F Clock schematic. Notice that the signal CRUSLCT- goes active (low) when the address 17EO (Hex) appears on the Address Bus. This signal remains low throughout the address range of 17EO to 17FF and goes to the Clock Module to enable addressing of the 15 bits which occupy this range. The actual addressing of the individual bits is done by the four least significant CRU address bits (A01 to A04).

2G-5. Bit Assignments

2G-6. Each of the 15 bits has two meanings to the CPU Module, one for an output (CROUT) operation and one for an input (CRIN) operation. Table 2G-1 shows the bit assignments for each bit.

2G-7. Clock Registers

2G-8. As shown in Table 2G-1, bits 0 through 3 function as digit select lines to the clock. Refer to the schematic again and notice that U12, the clock, has these lines (S0, S1, S2, S3) as inputs. The CPU Module can select 1 of 10 BCD digits (clock registers) for reading by setting these lines to the appropriate bit pattern. Table 2G-2 shows how digit selection is accomplished.

2G-9. OPERATIONAL SEQUENCES

2G-10. Overview

2G-11. Once the 1720A operating system has received the correct time from the user, it sets this time into the clock and sets the valid time flag (TVAL) to indicate that the clock has been set. From then on, as long as the battery back up is maintained, the operating system reads the correct time from the clock on each bootup. If for some reason the battery voltage drops to a value that would destroy the time information, the valid time flag (TVAL) is cleared by the hardware. Then the next time the operating system checks this flag, it will assume invalid data and return a relative time of zero which, when added to the reference date of 1-1-72, will give 00-00 1-1-72. In addition, a year increment bit (YINC) is set and used by the operating system to keep track of year rollover, as explained later.

2G-12. Setting the Time

2G-13. To set any of the five time periods (seconds, minutes, hours, days, months), the CPU Module sets the binary code for that digit into S0 through S3. Then it toggles (set high then low) the INCR bit. Following this, it reads the digit by setting the READ bit high and reading the D0 through D3 bits. It compares this value to the value needed and repeats the process until the digit is set to the correct value. Refer to the schematic again. Within the Clock Module hardware this sequence takes place as follows:

Table 2G-1. Bit Assignments

Bit #	Output	Input
0	S0 (digit select)	D0 (BCD digit)
1	S1 (digit select)	D1 (BCD digit)
2	S2 (digit select)	D2 (BCD digit)
3	S3 (digit select)	D3 (BCD digit)
4	INCR	Not used
5	Not used	Not used
6	READ	Not used
7	Not used	Not used
8	YR0 (year information)	YR0
9	YR1 (year information)	YR1
10	YR2 (year information)	YR2
11	YR3 (year information)	YR3
12	YR4 (year information)	YR4
13	YINC (year increment bit)	YINC
14	TVAL (valid time flag)	TVAL
15	Not used	ID (identification)

Table 2G-2. Clock Register Assignments

TRUTH TABLE	REGISTER (DIGIT) SELECTED	DATA BUS
S3 S2 S1 S0		D3 D2 D1 D0
0 0 0 0	None	1 1 1 1
0 0 0 1	None	1 1 1 1
0 0 1 0	Seconds Units	BCD
0 0 1 1	Seconds Tens	BCD
0 1 0 0	Minutes Units	BCD
0 1 0 1	Minutes Tens	BCD
0 1 1 0	Hours Units	BCD
0 1 1 1	Hours Tens = 1 AM	1 0 1 0
0 1 1 1	Hours Tens = 0 AM	1 0 1 1
0 1 1 1	Hours Tens = 1 PM	1 1 1 0
0 1 1 1	Hours Tens = 0 PM	1 1 1 1
1 0 0 0	Days Units	BCD
1 0 0 1	Days Tens = 0	1 1 1 1
1 0 0 1	Days Tens = Not 0	BCD
1 0 1 0	Month Units	BCD
1 0 1 1	Month Tens = 0	1 1 1 1
1 0 1 1	Month Tens = 1	0 0 0 1
1 1 0 0	None	1 1 1 1
1 1 0 1	None	1 1 1 1
1 1 1 0	None	1 1 1 1
1 1 1 1	None	1 1 1 1

1. Bits 0 through 3 (digit select) are set one at a time by placing the bit address on the Address Bus and initiating a CRU output sequence:

   a. CRUSLCT- goes active (low).

   b. CROUT is made high or low as necessary by the CPU Module.

   c. CRCLK - goes active (low pulse).

   d. U20 (D-type flip-flop) latches and holds the state of CROUT.

   e. The output of U20 acts as the data input to the two addressable 8-bit latches U11 and U1.

   f. A01 through A03 address a specific latch within both U11 and U1 (Q0-Q3).

   g. Pin 10 of U14 (1-of-4 decoder) goes active (low pulse) because A04 is low on pin 13 and CRCLK is a high pulse on pin 14. This low enables U11 and sets the addressed latch to the state of CROUT.

2. The INCR bit is toggled by the same process to increment the clock register that was addressed by bits 0 through 3.

3. The value of the addressed register is read by the CPU as follows:

   a. The BCD value of the register addressed previously during the CRU output operation is on the D0 through D3 outputs of the clock. These lines act as inputs to U13 (8-channel data selector).

   b. The CPU Module places the bit addresses for the bits 0 to 3 one at a time on the Address Bus and does a CRU input operation for each one.

      1. A4 is set low. This drives pin 4 of U14 low which enables U13.

      2. The address on A01 through A03 selects which of the D0 through D3 outputs of the clock are output to the CRIN line which goes to the CPU Module.

4. If the CPU Module finds that the value of the digit is not correct, it repeats the incrementing process until the value is correct.

2G-14. Setting the Year

2G-15. The operating system sets the year to the value given to it by the user into the Clock Module. To do this the CPU Module sets the Q0 through Q4 latches in U1 to the appropriate states. This information is available from then on as an output of U2. Notice that if A04 is set to a 1, U2 instead of U13 is enabled by the output of U14. The sequence within the Clock Module hardware is as follows:

1. Bits 8 through 12 (year information) are set one at a time by placing the bit address on the Address Bus and initiating a CRU output sequence:

a. CRUSLCT– goes active (low) upon address recognition.

b. CROUT is made high or low as necessary by the CPU Module.

c. CRCLK– goes active (low pulse).

d. U20 (D-type flip-flop) latches and holds the state of CROUT.

e. The output of U20 acts as the data input to the two addressable 8-bit latches U11 and U1.

f. A01 through A03 address a specific latch within both U11 and U1 (Q0-Q3).

g. Pin 11 of U14 (1-of-4 decoder) goes active (low pulse) because A04 is high on pin 13, and CRCLK is a high pulse on pin 14. This low enables U1 and sets the addressed latch to the state of CROUT.

2G-16. Checking for Valid Time

2G-17. When the operating system sets the time it also sets the Q6 output of U1 (TVAL) to a 1 (using the CRU output sequence described in Setting The Year). A low voltage monitor circuit consisting of U3, CR1 and associated components generates a clear signal to pin 2 or U1 upon power up if the battery voltage has dropped below the minimum allowable value. This resets the TVAL bit (Q6) to zero. The CPU can read the state of this bit from U2 (bit 14) and know if the time is valid.

2G-18. Checking for Year Rollover

2G-19. On July 1, the operating system sets YINC (bit 13 or Q5 of U1) to a 1 using the same CRU output sequence described in Setting The Year. Then, every time the 1720A is initialized, it tests this bit. If the month is January through June and YINC is still set then it knows that the 1720A has not been turned on previously in the new year, so it increments the year and sets YINC to 0. But, if the operating system finds YINC at 0, it knows that the preceeding procedure has already been done and it leaves YINC alone until July 1. This algorithm makes the assumption that the controller will be powered on at least once every 6 months.

2G-20. Reading the Time

2G-21. The CPU Module can read the time in months, days, hours, minutes and seconds, by following the read sequence outlined in Setting The Time. It can read the year by reading the state of the latches in U1 which act as data inputs to U2. The latches are read in the following sequence:

1. The CPU Module places the bit addresses for the bits 8 to 12 one at a time, on the Address Bus.

2. It does a CRU input operation for each addressed bit.

   a. A4 is set high. This makes pin 5 of U14 low which enables U2.

   b. The address on A01 through A03 selects which of the data inputs to U2 (bits 8 to 12) is output to the CRIN line to the CPU Module.

Section 2H
Optional Electronic Disk

2H-1. INTRODUCTION

2H-2. The optional E-Disk provides the 1720A with 64K words of memory which is word addressable via the memory mapped I/O process. This module has an auto-increment capability for its internal address register to provide faster throughput for a block of data. The 1720A can have up to two E-Disk Modules Installed.

2H-3. The E-Disk Module has three 8-bit registers through which the CPU Module controls all read and write operations. Table 2H-1 gives the address assignments for each of the two boards and Table 2H-2 gives information about the individual registers.

2H-4. OPERATIONAL SEQUENCES

NOTE

The address examples in the following paragraphs are for Board #1.

2H-5. Write Operation

2H-6. The CPU Module can store data in the E-Disk Module. The paragraphs that follow describe the necessary steps.

2H-7. First the CPU Module executes a memory mapped I/O Operation to send the storage address to the E-Disk. In this process, the CPU Module does a memory write operation, placing the address of the E-Disk Address Register (F5C0) onto the 1720A Address Bus and placing the memory address at which it wishes to store the data byte onto the 1720A Data Bus.

2H-8. Following this, the CPU Module executes a Memory Mapped I/O Operation to send the data to be stored to the E-Disk Module. It does this with a memory write operation, placing the address of the E-Disk Data Register (F5C4) onto the 1720A Address Bus and placing the data word it wishes to store onto the 1720A Data Bus.

2H-9. If there is another byte of data to be stored in the same block of continuous addresses within the E-Disk, the CPU module repeats the second step only. It can do this because the E-Disk automatically increments the address within its address register after each operation. This process continues until the CPU changes the contents of the address register with another Memory Mapped I/O Operation.

2H-10. Read Operation

2H-11. The CPU Module can read information from the E-Disk Module. The paragraphs that follow describe the necessary steps.

Table 2H-1. Electronic Disk Address Assignments

BOARD #	ADDRESS ASSIGNMENT
Board #1	F5C0 to F5C7 (Hex)
Board #2	F5C8 to F5CF (Hex)

Table 2H-2. Register Assignments

Reg.	Address Hex (BD# 1)	Read/Write	Name
1	F5C0	Write	Address
2	F5C4	Read/Write	Data
3	F5C6	Read/Write	STATUS/CONTROL

NOTE: In Reg. #3, on a status read, Bit 15 = Parity Error. On a Control write, Bit 15 = Parity Error INT ENABLE.

2H-12. First the CPU Module sends the address from which data is to be read to the E-Disk with a Memory Mapped I/O operation. It does this by doing a memory write operation, placing the address of the E-Disk Address register (F5C0) onto the 1720A Address Bus and placing the desired E-Disk memory address onto the 1720A Data Bus.

2H-13. Following this, the CPU Module reads the data with another Memory Mapped I/O Operation. To do this, it does a memory read operation, placing the address of the E-Disk Data Register onto the 1720A Address Bus.

2H-14. If there is another byte the CPU wishes to read from the next contiguous memory location, it repeats the second step only. The E-Disk automatically increments its address register after each operation.

2H-15. ADDRESS RECOGNITION

2H-16. Refer to the Electronic Disk Schematic to locate specific signals and components mentioned in this description. The Address Recognition and Register Decode circuits work together whenever the CPU accesses an E-Disk register. The Address Recognition circuit decodes 1720A Address Bus bits AD3 through AD15 to enable the Register Decode Circuit. The Register Decoder decodes bits AD1 and AD2 to select one of the registers in the module. The Register Decoder also decodes address bits from the Address Register and Multiplexer to produce the RAS (Row Address Strobes).

2H-17. In addition to enabling the Register Decoder, the output of the Address Recognition circuit (U40, pin-12) produces the ADACK- on the 1720A Control Bus and gates BUSWR - to the Data Buffers to establish the direction of data flow. If BISWR- is low, the Data Buffers act as receivers from the 1720A Data Bus. If BUSWR- is high, the Data Buffers act as drivers to the 1720A Data Bus.

2H-18. MEMORY ORGANIZATION

2H-19. The organization of the E-Disk memory is shown in Figure 2H-1. The two banks (Bank 0 and Bank 1) that make up the memory each consist of sixteen 32,768 X 1-bit dynamic RAMs for data storage and oneextra 32,768 X 1-bit dynamic RAM for parity storage. The RAMs in both banks have parallel address lines, however, as explained later, not all of the ICs accept a given address. Each of the 16 RAMs that make up a given memory bank has one of the data lines from the Data Bus as an input/output and the extra RAM has the parity bit. Therefore, each bank can hold 32K words of data, giving a total capacity of 64K words. Note that the parity bit goes to and from the Parity Circuit and is not a part of the Data Bus (DB).

2H-20. MEMORY ADDRESSING

2H-21. The CPU sends the address for a memory access to the Electronic Disk address register via the 1720A Data Bus. The following sequence of events takes place within the module:

1. In response to both the address on the 1720A Address Bus and to the low state of the BUSWR- signal from the 1720A Control Bus, the Module latches the 16-bit address from the 1720A Data Bus into the Address Register and Multiplexer (U8, U16, U9, U17).

2. The AD0 through AD13 outputs of the counter/latches act as inputs to the address multiplexer U4, and the AD14 and AD15 outputs go to the Register Decoder which decodes them into either RAS0 and RAS1 (Bank1), or RAS2 and RAS3 (Bank2), depending upon which memory bank the address falls into.

2H-22. MEMORY DATA TRANSFER

2H-23. As soon as the row address process is finished, the CPU does either a memory write or read to the Electronic Disk Data Register via the 1720A Data Bus. The CPU does this either to store or retrieve data at the address it sent in the previous step. The paragraphs that follow describe the process that takes place within the E-Disk.

2H-24. The Data Register Select output of the Register Decode block (U2 pin 6) goes active (high). (Note that the Data Buffer is a receiver if this is a write operation and a driver if this is a read operation. This status is controlled by the state of BUSWR-. BUSWR- also controls the state of the WRITE input to the memory ICs.) The state of RWE the RAS enable at U2, pin-15, and the CAS- output of the RAS-CAS Generator (U22, pin-6) are all coordinated by the RAS-CAS Generator which begins the process upon the assertion of Data Register Select (U2, pin-6). Notice that the output of the address multiplexer U4 can be either AD0 through AD6, or it can be AD7 through AD13, depending upon the state of its RWE input:

1. First RWE goes low to gate AD7 through AD13 out of U4 at the same time that the RAS enable to U2, pin-15 is active. This gates the row address into the memory bank that receives the RAS- strobes.

2. Next, RWE goes high to gate AD0 through AD6 out of U4, at the same time that CAS- goes active. This gates the column address into the memory chips.

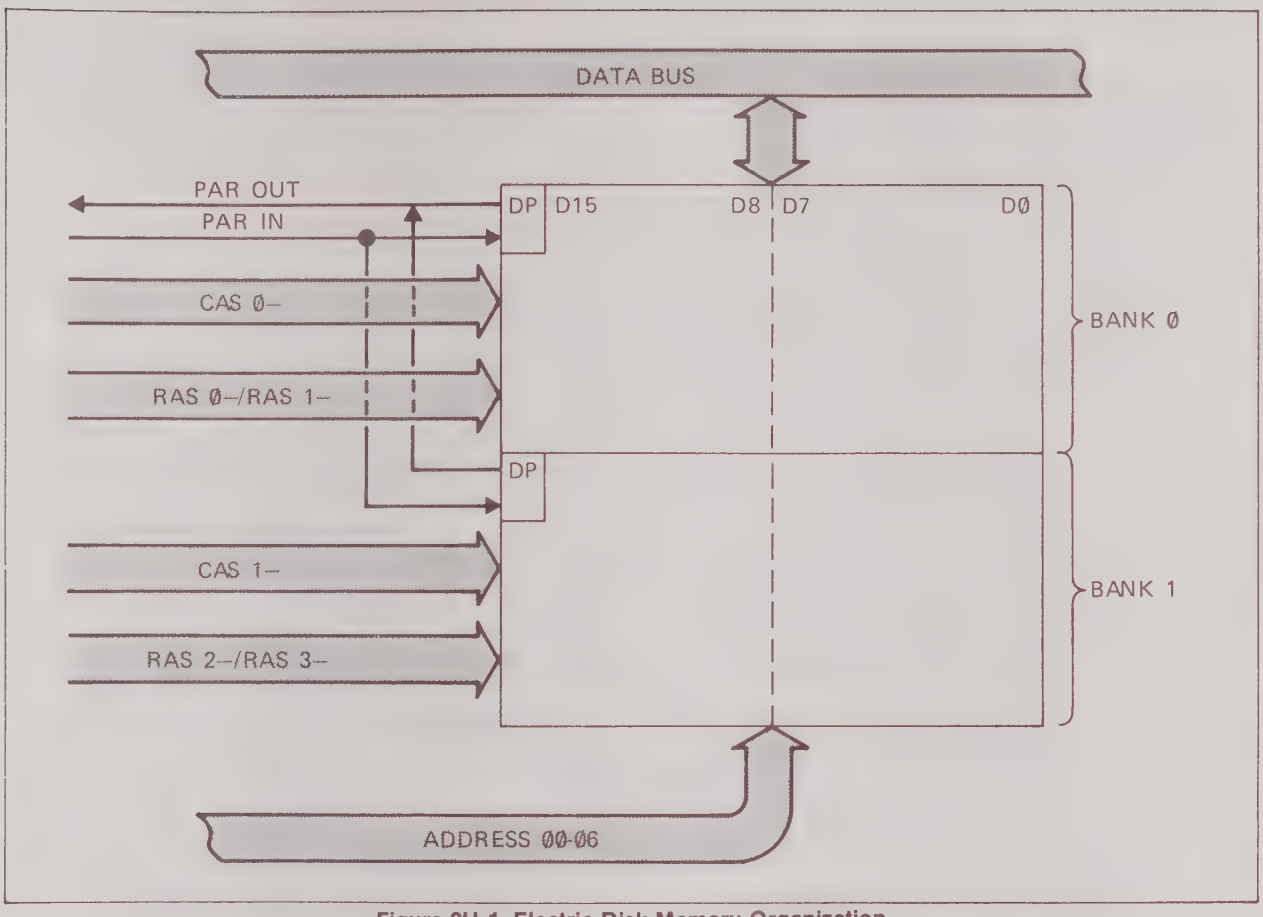


Figure 2H-1. Electric Disk Memory Organization

2H-25. Within the memory array, the ICs in the bank selected by the RAS strobes now contain the full 14-bit address (plus the correct RAS- strobe) necessary to access one of the possible 32K locations within that bank and actually begin the data transfer to or from the data bus.

2H-26. The actual data transfer takes place on the leading edge of CAS-, after the column address has been gated into the memory. The direction of transfer depends on the state of the WRITE input (determined by the state of BUSWR-). If WRITE is high the memory ICs store the data word from the data bus, and if it is low, they place the stored word onto the data bus.

2H-27. The Address Increment Control output, U72 pin-8, clocks the counter/latch ICs in the Address Register and Multiplexer to increment the address in the Address Register at the end of the operation.

2H-28. PARITY

2H-29. Refer to Figure 2H-2. The Parity Circuitry consists of ICs U25 and U33. They generate a parity bit during a write operation by making the PW line either high or low depending on the total number of high bits in the data word (high for even and low for odd). This produces an odd number of high bits in each memory location (including the parity bit). During a read operation, both the data word and the parity bit act as inputs to U25 and U33. U25 pin-5, goes high if there is not an odd number of high bits presented. This high output can produce an interrupt to the CPU Module if the CPU has enabled the interrupt (see Status/Control Register).

2H-30. MEMORY REFRESH

2H-31. The memory refresh description in the Memory Module theory of operation explains the need for refresh when using dynamic RAM ICs. It also explains how the refresh must be accomplished with RAS- only cycles during 2 ms time intervals. In the E-Disk Module, the Refresh Arbiter, Refresh Timer, and the Sync flip-flops all work together to perform the refresh of the memory ICs. They do this in two modes, battery back-up and normal, as explained in the following paragraphs.

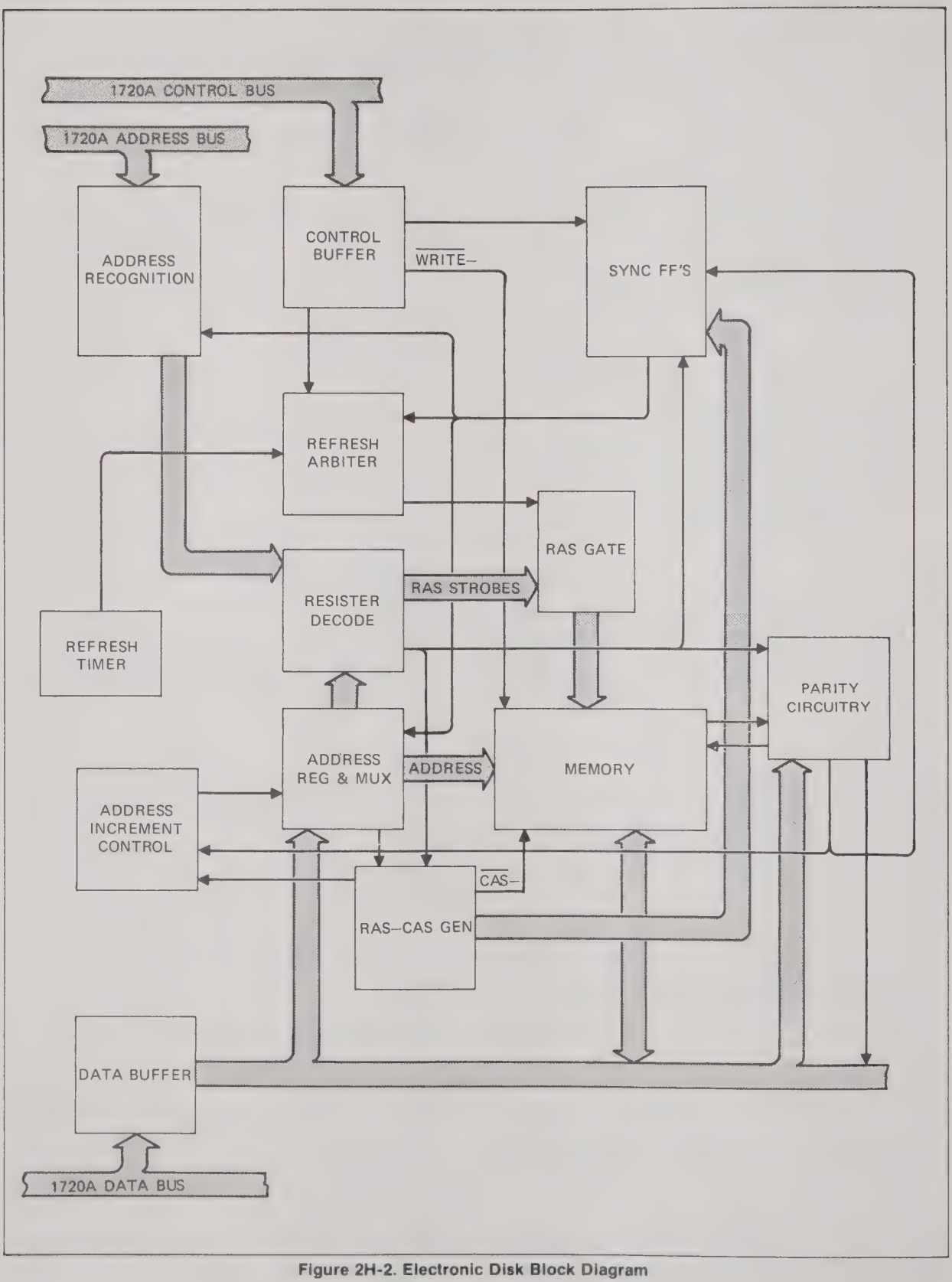


Figure 2H-2. Electronic Disk Block Diagram

2H-32. Normal refreshing occurs each time U31 pin-9, goes low. This low enables all four RAS outputs from the RAS Gate. The Refresh Arbiter produces this output each time the REFRQ output of the Refresh Timer (U14 pin-9) goes active (every 12 us) provided that the following conditions are met:

1. The system is not in battery back up (DCOK high).

2. The Module is not in the middle of a memory cycle (looks at the RAS flip-flop output).

3. The signal RINT- is not in the middle of the active low pulse. If this is the case, it will wait until the next RINT- and produce the output coincedent with the leading edge of RINT-.

4. The Refresh Arbiter circuit also eliminates certain undefined states in the flip-flop outputs from occurring as a result of the Refresh Timer being completely asynchronous with the system clock. These states, if they do occur, are not passed on to the memory chips.

2H-33. Each time the low pulse out of U31 pin-9 in the Refresh Arbiter circuit enables all four of the RAS outputs of the RAS Gate block, a refresh of one row address in each bank occurs. This address is an output of the Address Register and Multiplexer (U4). U4 has a RE (Refresh Enable) input (pin 2) and a CNT (Count) (pin 1). Inside this IC there is a counter, the output of which appears on the 00- through 06- outputs whenever the RE input is active (high). This counter counts upward each time the CNT input goes low, which happens each time there is a refresh. It cycles through all 128 row addresses, over and over again. As a result, a different address receives the refresh each time a refresh cycle occurs. Because the memory ICs do not receive a CAS- strobe during the refresh cycle, they do not perform a read or write operation.

2H-34. During battery back up, the Refresh Arbiter disregards all other considerations and refreshes one of the 128 row addresses every 12 us when the Refresh Timer output goes active (U14, pin-9). Notice that the RE input to U4 stays high at this time due to the BAT Sync flip-flop (U23, pin-9). The Refresh Arbiter still increments the internal counter of U4 with the CNT input each time it does a refresh.

2H-35. STATUS/CONTROL REGISTER

2H-36. As shown in the register assignment in Table 2H-2, the Status/Control Register performs a control function on a memory mapped write and a status function on a memory mapped read. The following paragraphs discuss these two functions.

2H-37. During a control function, the CPU sets DI15 to a 1 and does a Memory Mapped I/O write to address F5C6. Inside the Electronic Memory Module, DI15 and the control/status enable output of the register decode block (U2, pin-7) removes the set input to the parity error flip-flop (U49, pin-10). This leaves U49, pin-8 free to change state when a parity error detection occurs. This happens when there is a one output from the parity checker (U25, pin 5) during a memory read. This causes the output at U65, pin-11 to place a low on U49, pin-12. the clock for this flip-flop comes from the row enable flip-flop at the end of the memory cycle, and its output at pin-8 goes high. Remember, if the CPU does not remove the direct set input at U49, pin-10, there can be no change of states to indicate parity errors.

2H-38. During a status function, the CPU Module does a memory mapped read of the address of the Control/Status Register (F5C6). Any high output from U49 pin-8 would pass through U55 pin-5 to -6 and appear on DI15 to indicate a parity error.

2H-39. BATTERY OPERATION

2H-40. When the 1720A goes into battery back-up, most of the circuits in this module no longer receive +5V from the system. Only the refresh circuits receive their power from the battery-derived +5V and remain in operation to preserve the memory contents. Refer to the refresh description.

2H-41. BOARD SELECT SWITCH

2H-42. Refer to Table 2H-1, Electronic Disk Address Assignments. The switch S1 establishes the range of addresses a given Module responds to. Table 2H-3 details the function of the 4 individual switches within S1.

Table 2H-3. Electronic Disk Board Select Switch

S1				BOARD
1	2	3	4	ADDRESS RANGE (Hex)
off	off	off	off	F5C0 to F5C7
off	off	on	on	F5C8 to F5CF

Section 2J

PROGRAMMER KEYBOARD, TOUCH-SENSITIVE DISPLAY, AND CRT DISPLAY ELECTRONICS

2J-1. PROGRAMMER KEYBOARD

2J-2. Refer to the Keyboard Schematic. The keys on the keyboard are set up to form a matrix. The signals R0 through R7, representing the rows of this matrix, can only become active (high) when both a given key is pressed and the column in which that key is located is scanned. The microprocessor, located on the Video Keyboard Module, scans the columns by placing the appropriate BCD code on the S0 through S3 inputs of the 1-of-10 decoder IC. As a result, the keyboard output code on the R0-R7 lines consists of a binary bit (from the column being scanned) corresponding to the row in which the depressed key is located. The microprocessor in the Video Keyboard Module receives this code and any signal resulting from a SHIFT or CONTROL key depression. Then the microprocessor uses a PROM program to output ASCII code for transmission to the 1720A CPU Module.

2J-3. In addition to producing the ASCII code, the microprocessor also initiates a one-shot output from the Video Keyboard Module. This pulse enters the keyboard on pin 2 of the connector and activates a solenoid to make a click sound. This provides audible feedback to the keyboard operator.

2J-4. The microprocessor also produces the signals required to light the Caps Lock and Page Mode LEDs. It does this if it receives the code from either of these two keys.

2J-5. TOUCH-SENSITIVE DISPLAY

2J-6. Refer to Figure 2J-1. As shown, the Touch-Sensitive Display consists of two sheets of mylar which have vacuum deposited gold on one side. Scribe lines in this gold coating form electrically isolated conductive bands. The rear sheet is scribed horizontally and bonded to an acrylic backing panel. The front sheet is separated from the rear sheet by a gasket and is scribed vertically. This creates a 10 by 6 switch matrix with normally open contacts.

2J-7. If one of these grid switches is closed by an operator's touch, the microprocessor on the Video Keyboard Module produces a special code for output to the 1720A CPU Module. This output is a binary number from 1 to 60 (with the eighth bit set to 1) that corresponds to the switch position that was touched.

2J-8. CRT DISPLAY ELECTRONICS

2J-9. Refer to Figure 2J-2. The CRT display electronics produce a free running horizontal sweep (left to right) that scans downward on successive sweep lines, doing a horizontal retrace at the end of each line. When it reaches the bottom of the screen it does a vertical retrace to the top of the screen and repeats the process. This is done by passing appropriate sweep ramps through yoke coils on the neck of the CRT.

2J-10. As shown on the block diagram, the Horizontal and Vertical Sync inputs from the Video Keyboard Module pass through amplifier and shaper circuits to the oscillators that produce the sweep. In the presence of proper sync signals, these oscillators become synchronized to the video input which is also produced by the Video Keyboard Module.

2J-11. In response to this video signal, the Video Ampl./Output circuit in the display electronics modulates the trace into a series of dots and spaces. The Video Keyboard Module causes these dots and spaces to occur at the proper intervals to make characters appear on the display.

2J-12. The horizontal sweep output goes to the High Voltage Supply as well as the horizontal yoke coil of the CRT. In the High Voltage Supply, the sweep passes

through the primary coil of a high voltage transformer. This transformer has two secondaries. One secondary produces high voltage and the other secondary produces filament voltage for the CRT. The high voltage output is rectifed by a diode to provide DC for the CRT anode. The filament voltage goes directly to the CRT.

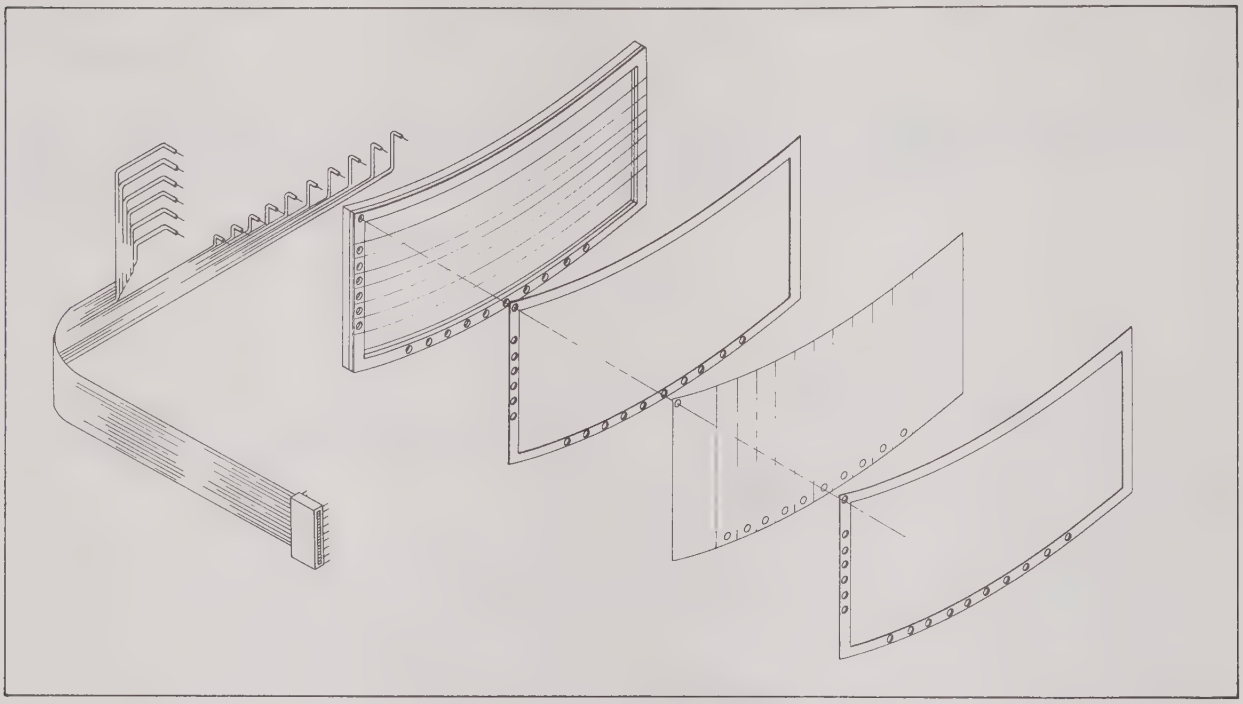


Figure 2J-1. Touch-Sensitive Display

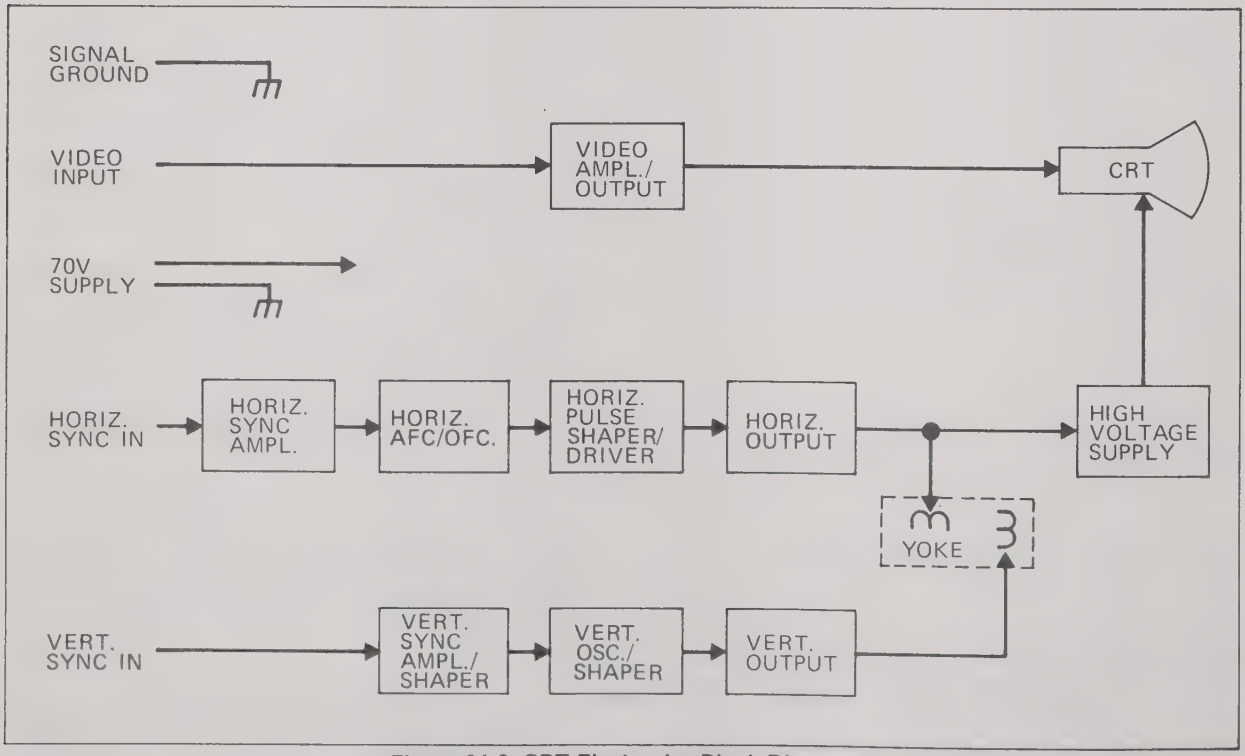


Figure 2J-2. CRT Electronics Block Diagram

Section 2K
Floppy Disk Drive Unit

2K-1. INTRODUCTION

2K-2. The Floppy Disk Drive Unit does the actual writing and reading of data to and from the floppy disk media. It does this under the direction and control of the DMA Floppy Module. This discussion covers both the drive unit and the track and sector format used on the media.

2K-3. THE DRIVE UNIT

2K-4. The Floppy Disk Drive Unit is fully self contained and requires no operator intervention during normal operation. It consists of a spindle drive system, a head loading and positioning system, and a read/write and erase system. Diskettes are inserted through the open front door. Media guides position the diskette laterally and up/down. A back stop ensures correct in/out positioning. Closing the front door activates the cone/clamp system. This system first centers the diskette then clamps it to the drive hub. The drive hub is driven at a constant speed of 300 rpm by a dc motor.

2K-5. In operation, the head load system loads the magnetic read/write head into contact with the recording medium. This is done by:

1. Referencing the diskette to a platen surface.

2. Referencing the head crown to the same platen surface as the diskette.

3. Providing a pressure pad which is loaded on the opposite side of the recording medium from the head with a force of 12 to 15 grams.

2K-6. A 4-phase stepper motor/cam assembly and associated electronic circuits positions the read/write head over the desired track. The positioner employs a 2-step movement to cause a 1-track linear movement.

2K-7. The drive has the following sensor systems:

1. A write protect sensor that disables the write electronics of the drive when a write protected diskette is inserted into the drive.

2. A Track 0 switch which senses when the read/write head is positioned over Track 0.

3. Index sensors, which consist of LED light sources and phototransistors that generate a digital pulse each time an index hole is detected.

4. A disk-present sensor.

2K-8. TRACK AND SECTOR FORMAT

2K-9. The 1720A must prerecord sector address information on the diskette before storing any information on it. This is done as a formatting process via a utility program before the disk is used by a user program. Each diskette has a single index hole which is used during this formatting process and for determining time-out (five revolutions without an address match). At the completion of the formatting, each of the tracks is divided into 10 sectors allowing data records to be stored under a track and sector address. These are not actual tracks (grooves) in the recording media. Instead, they are the places to which the read/write head positioner steps the head. The outermost track is track 00.

2K-10. Refer to Figure 2K-1. This simplified representation of the track and sector format shows the 10 physical sectors and the three outer tracks with the sectors numbered as they are addressed. Notice that the sectors are interleaved. This is done when the diskette is formatted by giving the sectors on each track an address in their ID field which differs from their physical location. In addition, the sectors are skewed four positions from track to track in order to compensate for the 25 ms step time.

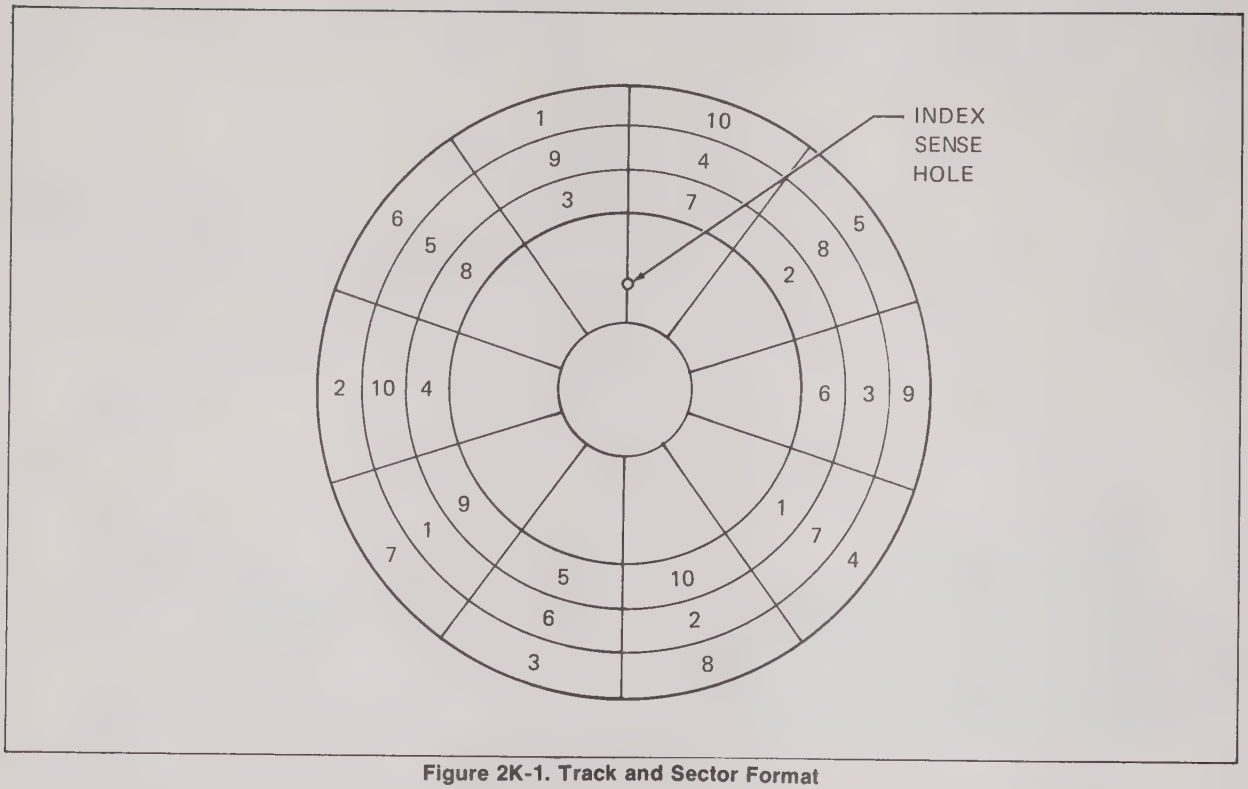
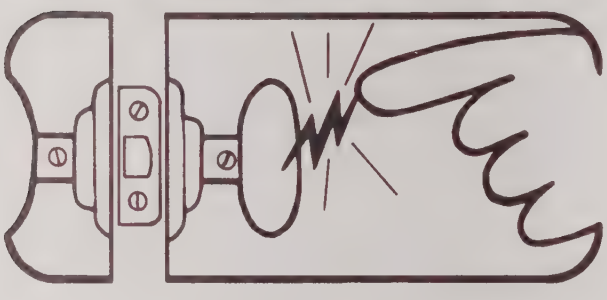


Figure 2K-1. Track and Sector Format

static awareness

A Message From

John Fluke Mfg. Co., Inc.

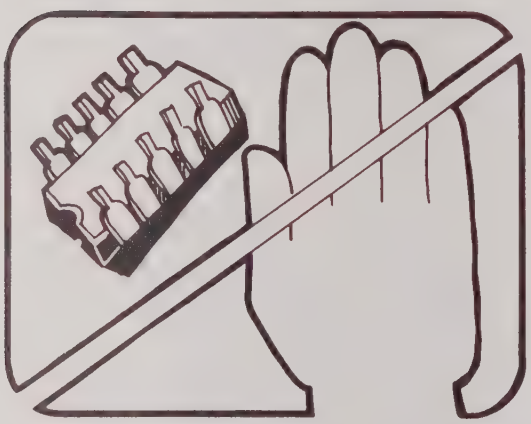


Some semiconductors and custom IC's can be damaged by electrostatic discharge during handling. This notice explains how you can minimize the chances of destroying such devices by:

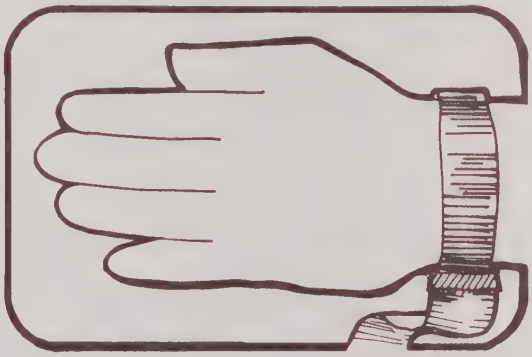
1. Knowing that there is a problem.
2. Learning the guidelines for handling them.
3. Using the procedures, and packaging and bench techniques that are recommended.

The Static Sensitive (S.S.) devices are identified in the Fluke technical manual parts list with the symbol "⊗"

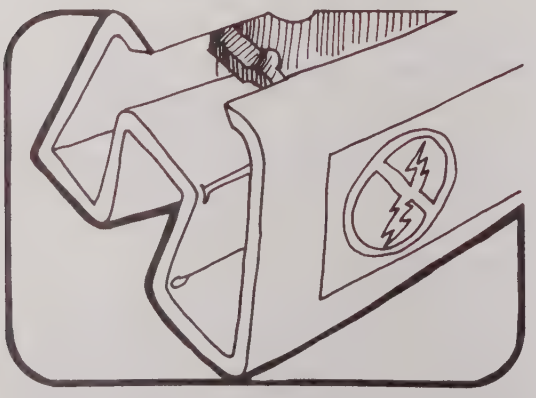
The following practices should be followed to minimize damage to S.S. devices.



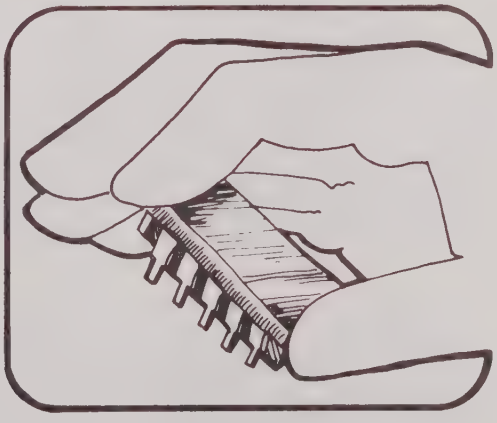
1. MINIMIZE HANDLING



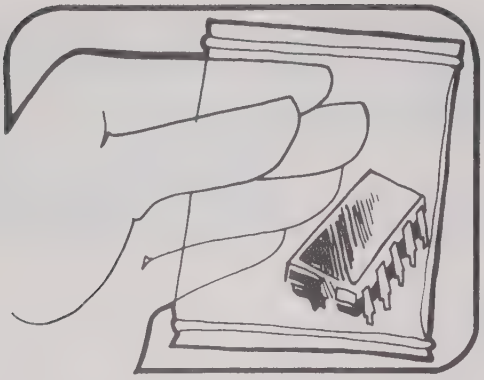
3. DISCHARGE PERSONAL STATIC BEFORE HANDLING DEVICES



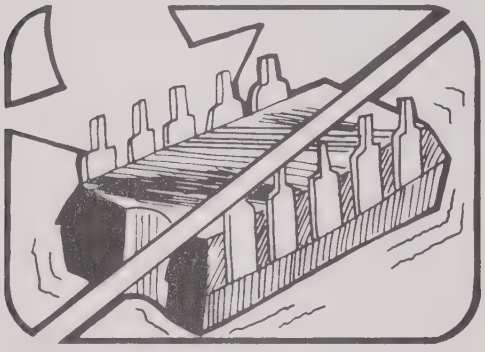
2. KEEP PARTS IN ORIGINAL CONTAINERS UNTIL READY FOR USE.



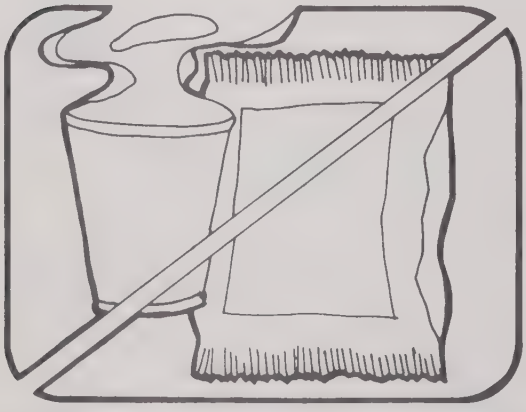
4. HANDLE S.S. DEVICES BY THE BODY



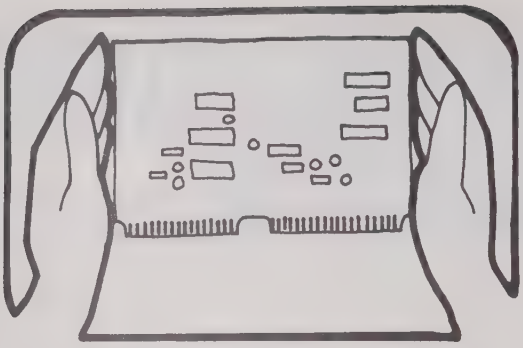
5. USE ANTI-STATIC CONTAINERS FOR HANDLING AND TRANSPORT



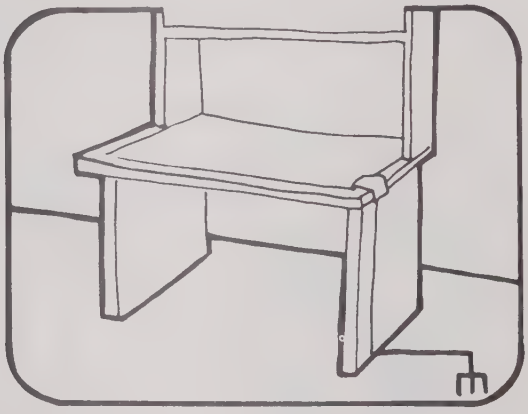
6. DO NOT SLIDE S.S. DEVICES OVER ANY SURFACE



7. AVOID PLASTIC, VINYL AND STYROFOAM® IN WORK AREA



8. WHEN REMOVING PLUG-IN ASSEMBLIES, HANDLE ONLY BY NON-CONDUCTIVE EDGES AND NEVER TOUCH OPEN EDGE CONNECTOR EXCEPT AT STATIC-FREE WORK STATION. PLACING SHORTING STRIPS ON EDGE CONNECTOR USUALLY PROVIDES COMPLETE PROTECTION TO INSTALLED SS DEVICES.



9. HANDLE S.S. DEVICES ONLY AT A STATIC-FREE WORK STATION
10. ONLY ANTI-STATIC TYPE SOLDER-SUCKERS SHOULD BE USED.
11. ONLY GROUNDED TIP SOLDERING IRONS SHOULD BE USED.

Anti-static bags, for storing S.S. devices or pcbs with these devices on them, can be ordered from the John Fluke Mfg. Co., Inc.. See section 5 in any Fluke technical manual for ordering instructions. Use the following part numbers when ordering these special bags.

Description	John Fluke Part No.	Price
6" x 8" Bag	453522	Per Current Parts Listing
8" x 12" Bag	453530	Per Current Parts Listing
16" x 24" Bag	453548	Per Current Parts Listing
12" x 15" Bag	454025	Per Current Parts Listing
Wrist Strap	571992	$7 00
30" x 60" x 60 Mil	RC-AS-1200	$22.00

PORTIONS REPRINTED
WITH PERMISSION FROM TEKTRONIX, INC.
AND GENERAL DYNAMICS, POMONA DIV.

® Dow Chemical

J0089B-07U7810/SE EN Litho in U.S.A.

Section 3
Maintenance

3-1. INTRODUCTION

3-2. The 1720A Controller has a modular design that mimimizes down time by allowing the user to diagnose troubles down to a module level. The diagnostic procedure presented at the beginning of this section has two parts: the Live Box Procedure and the Dead Box Procedure. These procedures can usually narrow the failure down to one of the replaceable modules. The Assembly/Disassembly Procedures that follow the Live and Dead Box Procedures describe how to remove and replace the faulty module. In addition, the user can quickly determine if the software or the hardware is at fault in any given problem by running the Live Box Procedure. If the system goes through the Live Box Procedure without any failures, then the problem must be caused by the software. The final portion of this section is the Preventative Maintenance Program which details the periodic actions necessary to maintain optimum performance.

3-3. System problems fall into two general catagories:

1. Live-box failures, where the display still functions and the system can load a disk.

2. Dead-box failures, where the display is not usable, and/or the system cannot load a disk.

3-4. To begin troubleshooting a problem, you must decide which type of failure you are attempting to troubleshoot. The only function of the Dead-box Procedure is to arrive at a point where the Diagnostic Disk can be loaded so that you can run the Live Box Procedure. Therefore, if the display is functioning and you can load the Diagnostic Disk, you should start with the Live Box Procedure. If this is not the case, you should start with the Dead Box Procedure.

NOTE

The locations of the test points, adjustments, and components called for throughout this section are shown in Figure 6-2 unless otherwise indicated. This is a fold-out figure and is located in Section 6 for ease of reference.

3-5. REQUIRED EQUIPMENT

3-6. Table 3-1 lists the required equipment. If the equipment recommended is not available, then other equipment with equivalent specifications may be used.

3-7. LIVE BOX PROCEDURE

3-8. Introduction

3-9. This procedure uses the CVFY (Controller Verification) program from the Diagnostic Disk to identify the faulty module. This program verifies that each module in the 1720A is operating correctly. The assumption is made that all hardware problems will cause a failure of at least one of the tests made by CVFY or by one of the other tests at the end of the procedure. The complete Live Box Procedure, therefore, constitutes a total verification of the system performance.

3-10. The CVFY program is an assembly language program that is loaded via the Console Monitor. It is a structured program that makes calls to modular subprograms. It calls these subprograms one at a time by loading them from the Diagnostic Disk then executing them. They return their pass/fail data to CVFY upon completion.

Table 3-1. Required Equipment

EQUIPMENT NOMENCLATURE	RECOMMENDED EQUIPMENT	FUNCTION/COMMENTS
LIVE BOX PROCEDURE		
Dual Trace Oscilloscope	Tektronix 465B	
IEEE Interface Cable	Fluke #Y8001	Connect IEEE ports together
RS-232-C Null-modem*	Fluke #Y1705	Connect RS-232-C Ports Together
Diagnostic Disk*	Fluke #521419	Diagnostic Program
DEAD BOX PROCEDURE		
Halt Button*	Fluke #580589	
24 Pin IC Puller*	Fluke #572339	
Alignment Tool*	Fluke #572321	
HDT PROMs*	Fluke (Odd) #576314 (Even) #576322	
Alignment Disk*	Fluke #572313	
Display Alignment Gauge*	Fluke #535864	
Jumpers (2)		10 inch, alligator
Dual Trace Oscilloscope	Tektronix 465B	
DVM	Fluke 8020A	
Customer Maintenance Kit	Fluke #577791	
* Customer Maintenance Kit (contains all * items above)		
Note: If your system does not make use of the RS-232-C ports, you can omit the RS-232-C Cable requirement. The test (#7) will record an error which can be ignored.		

3-11. On completion, each subprogram either displays data that can isolate the area of the fault to the nearest replaceable module or displays PASSED. You should watch the Display while CVFY is running. The subprograms that test the various cursor and display functions only attempt the various operations and rely on you to check that the attempt was successful and to keep track of any discrepancies between what the Display says will happen and what actually does happen. As soon as all the subprograms are finished, CVFY prints a summary indicating which subprogram passed or failed. Refer to Figure 3-1. As discussed in the Live Box Procedure Steps that follow, you will use this summary to locate the faulty module.

3-12. Live Box Procedure Steps

3-13. Before using this procedure read the following notes:

1. In the Live Box Procedure that follows, the term "enter" means type the indicated word or words then press RETURN.

2. ** indicates an evaluation or action step that directs you to take action or to go to another step within the procedure.

3. Anytime you replace a component in the system, you must run the Live Box Procedure to have assurance that the system is fully operational.

4. Whenever you are asked to press a control character, such as CTRL C, hold the CTRL key down while you press the indicated key.

5. The term "normal configuration" means that the 1720A is set up to run normally. In particular:

   a. All PCBs are in place.

   b. Regular PROMs are installed.

   c. All Normal/Test switches are in Normal.

   d. All covers, wires and cables are in place.

   e. No test jumpers are in the system.

   f. All connectors are installed.

6. Before calling Fluke Service, if this procedure fails to locate the fault, please go over all the steps you made that led up to that point in the procedure.

TEST SUMMARY TABLE

TIMES PASSED = 0	TIMES FAILED = 1	TIMES CYCLED = 1
TEST 1	CPU	PASSED
TEST 2	MEMORY	PASSED
TEST 3	VIDEO	PASSED
TEST 4	FLOPPY	PASSED
TEST 5	CLOCK	FAILED
TEST 6	IEEE	PASSED
TEST 7	RS232	FAILED
TEST 8	ELECTRONIC DISK #1	MISSING
TEST 9	TOUCH SENSE	PASSED

Figure 3-1. Sample CVFY Summary

STEP A. INITIAL SET UP

1. Install IEEE Interface Cable between the two ports.

2. Install RS-232-C Null Modem Adapter between the two ports.

3. Insert the Diagnostic Disk into the Drive Unit.

4. Press RESTART button.

5. The 1720A now displays the following:

   a. FLUKE 1720A CONTROLLER
      HELLO
      BOOT Vx.y

   b. FLUKE 1720A CONTROLLER
      SELF TEST

   c. FLUKE 1720A CONTROLLER
      LOADING

** If these messages do not appear, then go to the Dead Box Procedure.

STEP B. 1720A NOW LOADS CVFY

1. CVFY displays a menu and some instructions. Read the Display but take no action at this time. You'll notice that you may:

   a. Enter F for Final Test

   b. Enter A for Burn In

   c. Enter a test number to loop on an individual test, or enter a 0 to run all the tests.

2. This procedure assumes that you elect to run all tests, but you should be aware of two things:

   a. Test 9 (Touch Sense) requires operator interaction.

   b. The contents of the optional Electronic Disk memory are overwritten by Test 8.

3. After you enter 0, CVFY gives you a chance to catalog the results on the line printer (yes/no). Enter NO so that the results appear only on the display.

STEP C. 1720A NOW RUNS CVFY

1. CVFY calls the subprograms one at a time.

2. CVFY compiles a summary of the results.

3. CVFY displays the Test Summary.

STEP D. INTERPRETING CVFY RESULTS

1. The 1720A performance as a system depends on the correct operation of various functions. The tests that make up the CVFY program are designed to isolate and verify the performance of these functions. Some of the replaceable modules of the 1720A perform more than one function. Functions, however, do not occupy more than one module. Therefore, in general, failure of a specific test calls for replacement of the affected module.

2. Table 3-2 presents the catagories of functions tested by CVFY and relates them to replaceable

modules. All you need do is to locate the particular function that is failing in your system by looking on the CVFY Sample Summary, and use Table 3-2 to find which module you need to replace.

CAUTION

Turn the power off and turn the E-Disk Battery Disable Switch to the DISABLE position before removing or inserting any PCB or connector.

NOTE

Any time you replace a faulty module, you must do this part of the Live Box Procedure again. This verifies that the system is fully operational

STEP E. VERIFICATION OF DISPLAY ALIGNMENT

You should be entering this step from Step D and have already determined that the major 1720A system elements are performing correctly. This step can also be done as a stand alone verification of the display alignment at any time.

1. Load the Diagnostic Disk into the Drive.
2. Press RESTART
3. Wait while the 1720A loads the Diagnostic.
4. As soon as the CVFY Menu appears, press CTRL P.
5. As soon as the "#" prompt character appears, enter VIDEO.
6. Wait for the display to show you the video alignment pattern.

**The display now shows you the video alignment pattern. The purpose of this pattern is to show alignment of the display with respect to the touch-sensitive grid. Make sure that the pattern of brackets is aligned with the touch-sensitive grid. Parallax can cause good alignment to appear bad. Inspect the display from straight on. If the display is not aligned properly, go to the Video Display Alignment Procedure (paragraph 3-50) in the Assembly/Disassembly Procedures.

** If the display is in alignment, go on to part 7 of this step.

7. Press the CTRL P key to obtain the "#" prompt character and go on to the next step.

STEP F. VERIFICATION OF THE ENTIRE PROGRAMMER KEYBOARD

You should be entering this step from Step E, however, you can also use this step as a stand alone verification of the Programmer Keyboard at any time.

1. Load Diagnostic Disk
2. Press RESTART

Table 3-2. Interpreting CVFY Results

TEST	FUNCTION	MODULE	POSITION
1	CPU Instructions	CPU	1
2	Memory	Memory	3
3	Video	Video/Keyboard	6
4	Floppy Disk	See Note #1	2
5	Clock	DMA Floppy/Clock	2
6	IEEE-488	IEEE-488 Interface	7
7	Serial Data Transfer	CPU	1
8	Electronic Disk	Electronic Disk #1	4
9	Electronic Disk	Electronic Disk #2	5
10	Touch Sense	Display	--

Note: #1 A failure of Test 4 (Floppy Disk) can be a result of either a faulty DMA Floppy Module or a faulty Disk Drive unit. If this test fails, go through the Floppy Disk Drive Verification procedure that is a part of the Dead Box Procedure (Step J and K). This allows you to determine which of these two units is at fault.

Note: #2 The optional Electronic Disk modules have a switch that can make any given module #1 or #2. If for some reason you have only one of these modules and it is set up to be #2, the test would show module #1 failing.

3. Wait while the 1720A loads the Diagnostic.

4. As soon as the CVFY Menu appears, press CTRL P.

5. As soon as the "#" prompt character appears, press a key on the Programmer Keyboard and wait for the display to indicate the character pressed.

** If the display shows any character except the one you selected, then you must replace the programmer keyboard.

6. Continue pressing keys and observing the display until you have verified that all the keys on the Programmer Keyboard are operational.

STEP G. VERIFICATION OF FLOPPY DISK ALIGNMENT

This is a stand alone test of the Floppy Disk Drive. As part of the Live Box Procedure, it verifies the alignment of the Read/Write head in the Disk Drive and acts as a supplement to the CVFY Disk Test. Although this step is a part of the Live Box Procedure, which the Preventative Maintenance Program indicates should be done every 90 days, experience has shown that Disk Drive alignment is rarely a problem. Unless the system is having load problems, or giving some other indication of faulty Disk Drive performance, you may skip this step and still have reasonable assurance that the Disk Drive alignment is correct.

1. Insert the Diagnostic Disk.

2. Press RESTART.

3. Wait while the 1720A loads the Diagnostic.

4. As soon as the CVFY Menu appears, press CTRL P.

5. As soon as the "#" prompt character appears, enter QADSK0.

6. Turn the 1720A up on its side and remove the bottom cover. Locate the Floppy Disk Drive PCB. Use Figure 3-2 (Floppy Disk Drive PCB) throughout the following procedure to locate the various test points on the Disk Drive PCB.

7. Wait for the QADSK0 message. This test is self explanatory. Follow the directions that appear on the display. Refer to the Cat's Eye Alignment Waveform in Figure 3-3, if necessary.

** If the Disk Drive is out of alignment, it must be replaced. You run the risk of creating files that cannot be read on other units that are in alignment or on the current drive if it continues to drift out of alignment. Refer to the appropriate Assembly/Disassembly Procedure and replace the Disk Drive. Restore the system to its normal configuration and repeat this step.

3-15. DEAD BOX PROCEDURE

3-16. Introduction

3-17. As stated in the introduction to this section, the purpose of the Dead Box Procedure is to arrive at a point where the Live Box Procedure can be carried out. Therefore, the Dead Box Procedure assumes that you are faced with either an unusable display and/or a no-load condition. A no-load condition can result from a self-test error or from a hang up of the LOADING process during the start up routine.

3-18. Before using this procedure, read the following notes. Some of them protect the equipment from damage, and others assure that you understand the terminology used in the Dead Box Procedure Steps.

1. Turn power off before removing or inserting any PCB or connector.

2. ** identifies an evaluation and action step or a step that directs you to another part of the procedure.

3. Procedure steps are indicated by the word "Step" followed by a capital letter. Some procedure steps contain several lesser steps. These steps are indented and identified with letters or numbers. When a particular step is referred to, commas separate the designators. For example: Step J, 8, c indicates the c part of the eighth step within Step J.

4. The term "normal configuration" means that the 1720A is set up to run normally. In particular:

   a. All PCBs are in place.

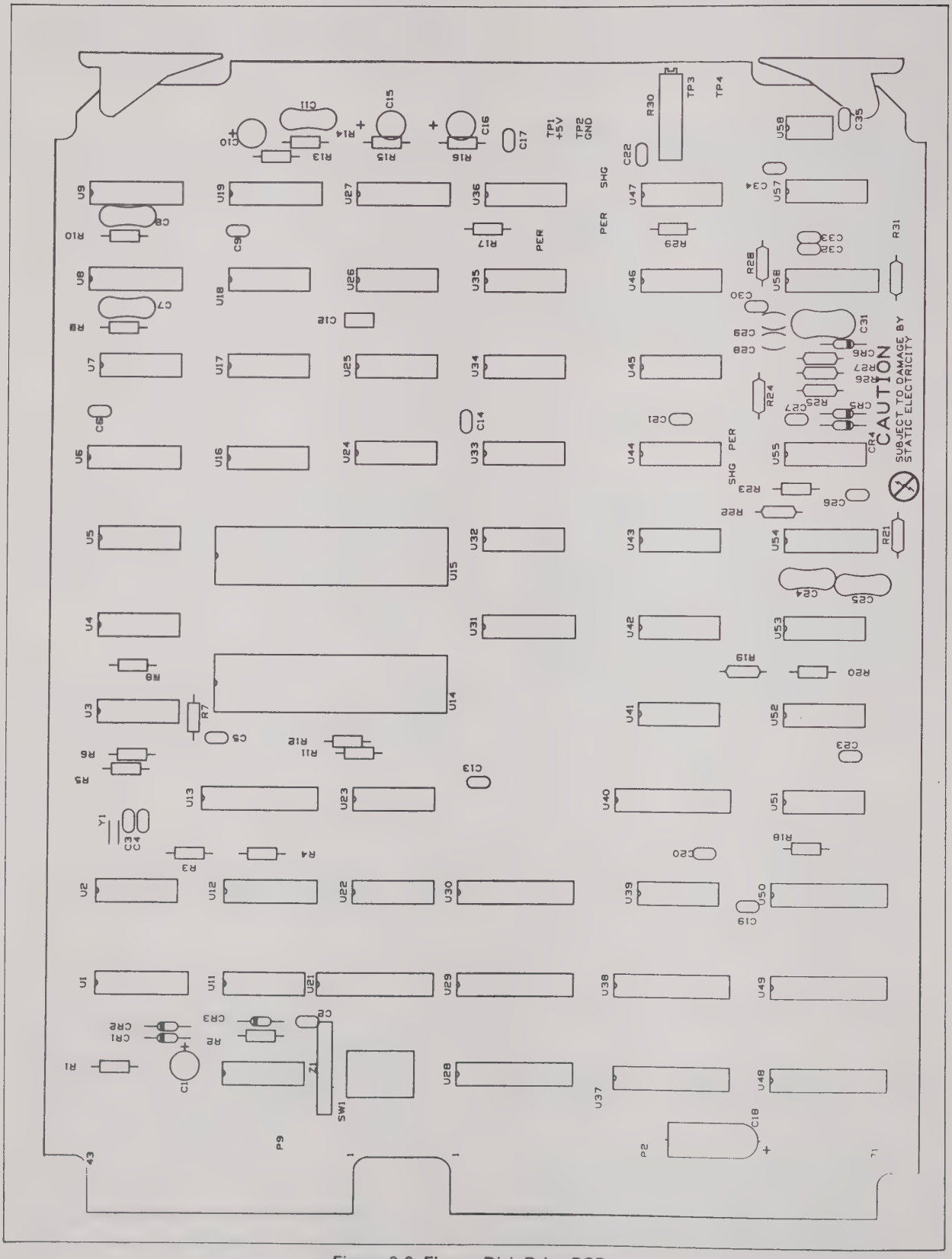


Figure 3-2. Floppy Disk Drive PCB

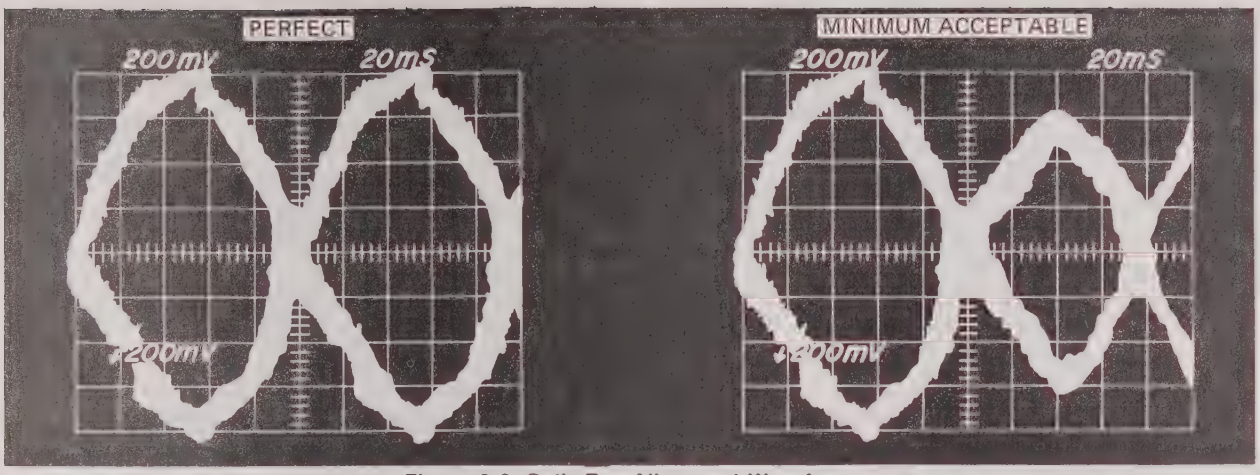


Figure 3-3. Cat's Eye Alignment Waveform

b. Regular PROMs are installed

c. All Normal/Test switches are in Normal.

d. All covers, wires and cables are in place.

e. No test jumpers are in the system.

f. All connectors are installed.

5. Unless otherwise noted, all measurements are made using TP 1 on the Mother Board as a ground reference.

3-19. Dead Box Procedure Steps

STEP A. START

1. Do not insert a disk until the procedure calls for it.

2. Plug the 1720A in and turn the power on.

** If there is no display and the fan is not running (to act as a power on indicator), then you must insure that:

a. Line power is correct.

b. Rear panel power-line fuse is the correct size and not open.

c. The Power Supply has not tripped due to overheating.

d. POWER switch is on.

STEP B. POWER SUPPLY VERIFICATION

The purpose of this step is to determine if the Power Supply is at fault.

1. Turn the power off.

2. Remove the top cover from the instrument.

3. Turn the power on.

4. Refer to Figure 3-4 and verify that all the indicated voltages and the Control Bus signals DCOK and ACOK are present. Check DCOK at TP5 (approximately 3.5V) and check ACOK at P14 pin-32 on the Power Supply Connector (approximately 3.5V).

** If any voltage or signal cannot be verified then go on to Step B.

** If all the voltages can be verified, then go on to Step C.

5. You should be entering this step from Step B, 4; one or more of the Power Supply voltages or signals could not be verified. Part a covers low or missing voltages or signals, and Part b covers out of tolerance voltages that can be adjusted.

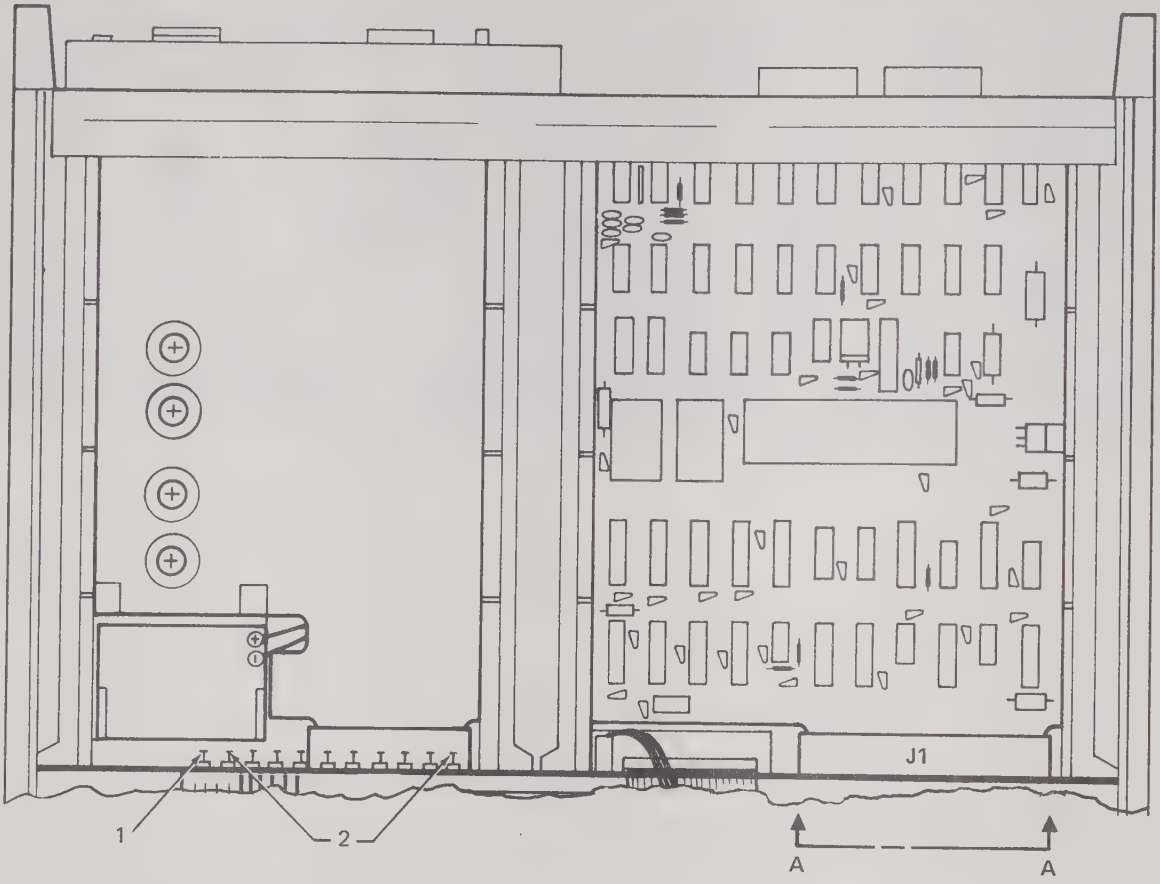
a. Low or missing voltages or signals:

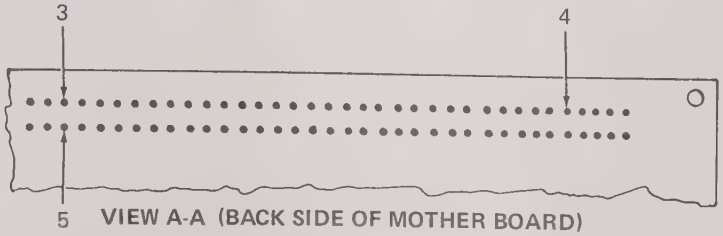
1. Turn power off.

2. Remove the Disk Drive power connector.

3. Remove the CRT Electronics Assembly edge connector.

4. Turn power on.





VIEW A-A (BACK SIDE OF MOTHER BOARD)

ITEM NO.	NAME
1	TP11 +70V Return
2	TP1 Common for all voltages except 70V TP2 +5V TP3 +12V TP4 Battery 6.8V TP5 DCOK TP6 −12V TP10 +70.0V with respect to TP11
3	+5VB (pin 5 on J1 Mother Board)
4	−5VB (pin 63 on J1 Mother Board)
5	+12VB (pin 6 on J1 (Mother Board)

NOTE: ALL VOLTAGES ARE ±5% EXCEPT 70V

Figure 3-4. Power Supply Verification

5. Check the Power Supply voltages, DCOK and ACOK as before. The 70V supply should now read approximately 115V with respect to TP11 because the CRT Electronics have been disconnected.

** If the voltages are still incorrect, turn power off, reconnect both connectors and go on to Step B, 5, 7.

** If the voltages are now correct, one of these two assemblies is faulty. Turn power off and re-connect one of the connectors. Turn power on and re-check the Power Supply Voltages, DCOK and ACOK as before. If the voltages are now incorrect, replace the faulty assembly; restore the system to its normal configuration, then run the Live Box Procedure.

6. Repeat this process with the other connector.

7. Turn the E-Disk Battery Switch to the DISABLE position.

8. Remove one of the PCB Modules. Refer to the Assembly/Disassembly Procedures if you do not know how to do this. Notice that if the CPU Module is the one removed, the +12V supply should read approximately 14V and DCOK and ACOK read approximately 1.5V. The other Modules should leave the readings unchanged, if they are removed one at a time, as detailed in the following steps.

9. Recheck the supply voltages.

** If the voltages are now correct then the Module you removed is faulty. Replace it, restore the system to its normal configuration, and run the Live Box Procedure.

** If the voltages are still incorrect, and you have not removed all the Modules one at a time: turn the power off; replace the Module you removed; remove a different module, and turn the power on. Go back and recheck the supply voltages (step 9).

** If the voltages are still incorrect and you have removed all the Modules one at a time, and confirmed that none of them was loading down the Power Supply, then the Power Supply is faulty. Replace it; restore the system to its normal configuration, and run the Live Box Procedure.

b. Out of tolerance voltages that are within adjustment range:

NOTE

All other voltages are slaved to the 5V supply. Adjusting the 5V supply may solve the problem.

1. Monitor TP10 with a DVM. Adjust the 5V adjustment pot until TP10 measures 70.0V.

2. Check that all other supplies are in tolerance.

** If this does not work, the Power Supply may be faulty. Go to part A of this step.

** If this does work, run the Live Box Procedure.

STEP C. SELF TEST AND LOAD

1. Insert the Diagnostic Disk into the Disk Drive.

2. Press RESTART. The system now attempts to run the Self Test program from PROM. On a self test error, it will ask you to "Press any key..." in order to continue.

** If the display remains blank or has a bright background raster, or if no message appears, then go to Step D.

** If there are any Self Test Errors, go to Step D.

3. The system now attempts to run the Load program from the PROM and tries to load the Diagnostic Disk.

** If the word "LOADING" appears, but the load does not finish (no Menu of test options), try a backup Diagnostic Disk (go back to Step C, 1). If you have already tried a new disk, go to Step J.

** If the Menu appears, then remove the Diagnostic Disk and press restart. This forces an error condition (Disk Not Mounted) and the program asks you to 'Press any key to continue'. Press any key on the Programmer Keyboard at this time.

** If the Self Test does not respond (flash the Display) when you press the key, then unplug the Programmer Keyboard and press the Touch-Sensitive Display.

** If the Self Test responds then the Programmer Keyboard must be replaced.

** If the Self Test does not respond, then you have to replace the Video Module. Refer to the appropriate Assembly/Disassembly Procedure; replace the Video Module; restore the system to its normal configuration and run the Live Box Procedure.

** If you tried a backup Diagnostic Disk and the Self Test responded (flashed the Display) when you pressed the key, then you know the Diagnostic Disk

loaded and that your Programmer Keyboard works. Run the Live Box Procedure. If it passes, the original disk you were trying to load is bad.

STEP D. VIDEO VERIFICATION

You should be entering this step from Step C and should have verified that the Power Supply is not at fault. The purpose of this step is to verify that the Video/Keyboard Module (Video Module) is operating correctly so that you can use it in succeeding steps to verify the operation of the other system elements.

1. Turn system power off.

2. Remove the Video Module, referring to the Assembly/Disassembly procedure if necessary, and set the Test/Normal Switch to Test. Reinstall the Video Module.

3. Turn the Power on.

4. Use the Programmer Keyboard and type in a character. Hold the key down until the character completely fills the display. Repeat for all characters.

** If characters are normal then the Video Module is performing correctly. Go on to Step E.

** If characters appear but have a vertical roll, then go to Step M.

** If characters appear but have a horizontal roll, as evidenced by a horizontal drift or a tearing effect, go to Step N.

** If the background raster is visible, covering the entire display area with light, go to Step 0.

** If no characters appear, then turn the Brightness Pot on the rear edge of the Video Module PCB fully CW. If no characters appear, turn the pot back to where it was and go on to Step D,5. If they do appear, adjust the pot for normal viewing and run the Live Box Procedure.

5. If characters do not appear on the display, the fault can be either the Video PCB, the CRT Electronics, or the CRT. Proceed as follows to isolate the fault:

a. Turn the power off.

b. Remove the Video Module from the system.

c. Turn power on.

d. Attach a jumper between TP5 and TP7 on the Mother Board.

** If a full, bright, flickering raster appears, then the Video Module is bad. Replace it; restore the system to its normal configuration, and run the Live Box Procedure.

** If the display is other than a full, bright flickering raster, remove the jumper and go to Step L.

STEP E. CPU BOARD VERIFICATION

You should be entering this step from Step D. You have already verified that the Video Module is operating correctly and are now ready to use it in conjunction with the Self Test PROM to verify the operation of the other system elements.

1. Turn power off.

2. Remove the Video Module; set the Normal/Test switch to Normal, and put the module back into the system.

3. Turn power on.

The display should show self test errors for only:
Memory
Floppy
IEEE

** If this condition does not occur, the CPU Module PCB is bad. Replace it and try Step E again.

** If this condition does occur, then go on to the next step.

STEP F. DMA/FLOPPY BOARD VERIFICATION

1. Turn power off.

2. Insert the DMA/Floppy PCB.

3. Turn power on.

The display should show self test errors for only:
Memory
IEEE

** If this condition does not occur, the DMA/Floppy Module PCB is bad. Replace it and try Step G again.

** If this condition does occur, go on to the next step.

STEP G. MEMORY BOARD VERIFICATION

1. Turn power off.

2. Insert the Memory PCB.

3. Turn power on.

The display should show self test errors for only: IEEE

** If this condition does not occur, the Memory PCB is bad. Replace it and try Step G again.

** If this condition does occur, go on to the next step.

STEP H. IEEE-488 BOARD VERIFICATION

1. Turn power off.

2. Insert the IEEE-488 PCB.

3. Turn power on.

The display should show no self test errors.

** If this condition does not occur, the IEEE-488 Module is bad. Replace it and try Step H again.

** If this condition does occur, go on to the next step.

STEP I. OPTIONAL ELECTRONIC DISK BOARD VERIFICATION

1. Turn power off.

2. Insert Electronic Disk PCB #1.

3. Turn power on.

The display should show no self test errors.

** If this condition does not occur, the Electronic Disk PCB is bad. Replace it and try Step I again.

** If this condition does occur, go on to the next step.

4. Repeat with Electronic Disk #2.

STEP J. FLOPPY DISK DRIVE VERIFICATION (READ)

If you are entering this step and can load the Diagnostic Disk (see Note #1 in Table 3-2) then go directly to Step J, 13. If you cannot load a disk, then start at Step J, 1.

1. Turn the power off.

2. Turn the Controller on its side and remove the bottom cover.

3. Turn power on.

4. Refer to Figure 3-2 (Floppy Disk Drive PCB) and verify that all voltages are within specified limits.

** If the voltages are correct, go to the next step (Step J, 5).

** If any of the voltages are outside of the specified limits then the fault is in one of the various connections between the Power Supply and the Floppy Disk Drive unit. Inspect the Floppy Disk Drive unit power connector and the solder connections to the Mother Board.

** Note any broken wires, connector pins, or faulty connections, then call Fluke Service.

5. Install the HALT button in the Power Supply rear connector (P25).

CAUTION

Before going on to the next step, refer to the appropriate Assembly/Disassembly Procedure to install the HDT PROMs.

6. Install the HDT (Hex Debugging Tool) PROMs, and observe the following guidelines as you perform the rest of the procedure:

a. The term "enter" means type the indicated expression on the Programmer Keyboard exactly as shown. An entry is terminated by either a Carriage Return or a Space Bar, as noted. If you make a mistake on any given entry you must press HALT and start over again from the beginning of the series of entries within that step. Be sure that you have the "." prompt character before any entry, especially when starting over. If its not there, press HALT to get it.

b. (SP) indicates the Space Bar.

c. (CR) indicates the Carriage Return key.

d. Lower case N (n) indicates that the display can be any digit or the letters A through F.

e. The ** symbol indicates an evaluation step that steers the operator to other procedural steps.

f. A representation of what the display should look like appears at the end of each step where you have to make these entries. Check your work against this to spot any errors you may have made.

7. Insert an alignment disk and press RESTART. You should get a display as shown below:

```
HDT V1.n.
```

** If this does not work, your HDT PROM set is bad. Try another set.

** If you do get the "." prompt character then go on to the next step.

8. Verify operation of the write protect and disk insertion switches as follows:

a. Set up the scope:

VERT----2 V/cm, DC coupled
HORZ----1 ms/cm
TRIG----Auto (free run)

b. Place the scope probe on U3 pin-12 and observe the level change as a write protected diskette is fully inserted and then withdrawn.

** The level should be >3V (enable) with the diskette fully inserted and should drop to <0.7V (disable) with the diskette out of the unit. If these conditions cannot be met, the Floppy Disk Drive Unit must be replaced. Refer to the appropriate Assembly/Disassembly Procedure and replace the Floppy Disk Drive. Restore the system to its normal configuration, then run the Live Box Procedure.

9. Insert an alignment disk, and press the "B" key on the Programmer Keyboard.

** You should hear a click sound from the Floppy Disk Drive Unit as the head loads. If you do not hear a click, go to Step J, 10. If you do hear a click, then go to Step J, 13.

In addition to the click, the display should show:

```
HDT V1.n

Loading
```

10. Press RESTART

11. Use HDT to load the head by making the following entries:

Enter	HDT Displays
SF3E8(SP)	F3E8 nnFF-
FFFF(CR)	.
SF3E8(SP)	F3E8
00FF(CR)	.

You should hear a click from the Floppy Disk Drive Unit when you press the last RETURN.

** If you do hear a click, then it means that you have a problem in your CPU Module. Refer to the appropriate Assembly/Disassembly Procedure and replace the CPU Module. Restore the system to its normal configuration then run the Live Box Procedure.

** If you do not hear a click, then go on to the next step.

```
HDT V1.n
 SF3E8
F3E8 nnFF-FFFF
.SF3E8
F3E8 nnFF-00FF
```

Display at end of Step J-11

12. Use HDT to isolate a Drive Unit fault from a Module fault as follows:

a. Leave the scope as in (Step J 8,a). Refer to Figure 3-2. Look at U2 pin-10 on the Floppy Disk Drive Unit PCB.

b. Press RESTART then use HDT to load the head by entering:

Enter	HDT Displays
SF3E8(SP)	F3E8 nnFF-
FFFF(CR)	.
SF3E8(SP)	F3E8 nnFF
00FF(CR)	

** The level on U2 pin-10 should be >3V at the start and should drop to <0.7V at the end. If either of these two conditions are not met, the DMA/Floppy Module is bad. Refer to the appropriate Assembly/Disassembly Procedure; replace the module, and run the Live Box Procedure.

** If both conditions can be met, continue with this step.

HDT V1.n
.SF3E8
F3E8 nnFF-FFFF
.SF3E8
F3E8 nnFF-00FF

Display at end of Step J, 12, b

c. Wait at least 25 seconds (allows a time out to occur).

d. Using the existing scope settings, place the scope probe on U2 pin-6. You should see >3V dc.

e. Press HALT.

f. Make the following entries:

Enter	HDT Displays
SF3E0(SP)	F3E0 nnnn-
300(CR)	.

** The level on U2 pin-6 should be >3V at the start and <0.7V at the end. If either of these two conditions cannot be met, the DMA/Floppy Module is bad. Refer to the appropriate Assembly/Disassembly Procedure and replace the DMA/Floppy Module. Restore the system to its normal configuration, then run the Live Box Procedure.

** If both the conditions can be met, the Floppy Disk Drive Unit is bad. Refer to the appropriate Assembly/Disassembly procedure and replace the Floppy Disk Drive Unit. Restore the system to its normal configuration, and run the Live Box Procedure.

Halt
.SF3E0
.F3E0 nnnn-300
.

Display at end of Step J, 12, f

13. Use HDT to verify the Floppy Disk Drive Radial Track Alignment.

a. Press RESTART.

b. Insert the Alignment Diskette into the Floppy Disk Drive Unit.

c. Refer to Figure 3-2 which shows the Floppy Disk Drive PCB, and use the scope as follows:

1. Attach the scope signal probes to TP2 and TP5. Connect the probe ground leads to TP11.

2. Attach the scope trigger probe to TP4. Connect the probe ground lead to TP1.

3. Set up the scope:

VERT----200 mV/cm, dual trace, A+B, with B inverted
HORZ----20 ms/cm
TRIG----External

d. Enter a program as follows:

Enter	HDT Displays
S100(SP)	0100 nnnn-
2E0(SP)	nnnn
1000(SP)	nnnn-
4E0(SP)	nnnn
F3F0(SP)	0108 nnnn-
10FD(CR)	.
SF3E8(SP)	nnnn-
FFFF(CR)	.
SF3E8(SP)	nnnn-
00FF(CR)	.
SF3E0(SP)	nnnn-
300(CR)	.
SF3E6(SP)	nnnn
10FF(CR)	.
SF3E0(SP)	nnnn-
1700(CR)	.
G100(CR)	.G100

** The CPU begins executing the program. This program is a loop that causes the Floppy Drive Unit to do a continuous read of track 16 on the Alignment Diskette. Adjust the scope triggering to obtain the waveform shown in Figure 3-3 (Cat's Eye Alignment Waveform) and complete the evaluation as follows:

1. Measure the amplitude of side A and of side B.

2. Compare A to B. The smaller of the two must be at least 80% of the larger.

3. If this condition is met, the Drive Unit performance of several parameters is verified: Head movement from track 00 to track 16, Read ability, and alignment.

** If this condition is not met, the Floppy Disk Drive Unit is bad. Replace it and run Live Box Procedure.

```
HDT V1.n
.S100
0100 nnnn-2E0 nnnn-1000 nnnn-4E0
nnnn-F3F0
0108 nnnn-10FD
.SF3E8
F3E8 nnnn-FFFF
.SF3E8
F3E8 nnnn-00FFF
.SF3E0
F3E0 nnnn-300
.SF3E6
F3E6 nnnn 10FF
.SF3E0
F3E0 nnnn-1700
.G100
```

Display after Step J, 13,d

14. Use HDT to move the head and check for head movement ability.

a. Do not change any of the set up established in Step J, 13.

b. Press HALT

c. Enter the following:

Enter	HDT Displays
SF3E6(SP)	F3E6 nnnn-
24FF(CR)	.
SF3E0(SP)	F3E0 nnnn-
1700(CR)	.
G100(CR)	

** The Cat's Eye Waveform on the scope should go away as soon as the Carriage Return is complete because head moves off the alignment track (16) and goes to track 32. If the waveform does not go away, the Floppy Disk Drive Unit is bad. Replace it and run the Live Box Procedure.

```
HALT
.SF3E6
F3E6 nnnn-24FF
.SF3E0
F3E0 nnnn-1700
.G100
```

Display at end of Step J, 14, c

d. Press HALT then enter the following:

Enter	HDT Displays
SF3E6(SP)	F3E5 nnnn-
10FF(CR)	.
SF3E0(SP)	F3E0 nnnn-
1700(CR)	.
G100(CR)	

** The Cat's Eye waveform should come back again. If it does not come back, it means that the head cannot move from track 32 to track 16. Replace the Floppy Disk Drive Unit and run the Live Box Procedure.

```
HALT
.SF3E6
F3E6 nnnn-10FF
.SF3E0
F3E0 nnnn-1700
G100
```

Display after Step J, 14, d

15. Final read data output verification

a. Allow the Drive Unit to continue reading the alignment track, as in Step J, 14, d.

b. Set up the Scope:

VERT----2 V/cm, single trace, DC coupled
HORZ----1 us/cm
TRIG----Internal

c. Refer to Figure 3-2. Place the scope probe on U3 pin-3.

d. Verify that the correct waveform is present at U3 pin-3. The waveform should go from 0V to +5V. The period should be 7.5 us from the leading edge of one positive-going pulse to the leading edge of the next one. The pulse width of the negative-going portion of the waveform should be 1 us. These values are typical and are given for troubleshooting purposes only.

** If the waveform cannot be verified, the Floppy Disk Drive Unit is bad. Replace it, and run the Live Box Procedure.

16. Drive motor speed verification

a. Allow the Drive Unit to continue reading the alignment track, as in Step J, 14, d.

b. Refer to Figure 3-2 which shows the Floppy Disk Drive PCB, and use the scope as follows:

1. Place the probe on U3 pin–6.

2. Set up the scope:

VERT----2 V/cm, single trace, DC coupled
HORZ----50 ms/cm
TRIG----Internal

c. Verify that the waveform is a 4 ms negative-going pulse (5V to 0V) with a repetition rate of 200 ms.

d. Connect a counter-timer to TP4 and measure the duration of the Index to Index pulse. The duration must be 200 ms ± 3 ms.

** If the waveform or timing cannot be verified, the Floppy Drive Unit is bad. Replace it and run the Live Box Procedure.

17. Verification of the track 00 detect switch

a. Press HALT

b. Set up the scope:

VERT----2 V/cm, single trace DC coupled
HORZ----1 ms/cm
TRIG----Auto

Place the probe on U3 pin–8. Since the head is not at track 00, the voltage should be >3V.

** If this condition is not met, replace the Floppy Disk Drive. Refer to the appropriate Assembly/Disassembly Procedure. Restore the system to its normal configuration and run the Live Box Procedure.

c. Enter the following:

Enter	HDT Displays
SF3E0(SP) 300(CR)	F3E0 nnnn-

d. The voltage on U3 pin–8 should drop to <0.7V as soon as the head moves to track 00.

** If this condition is not met, replace the Floppy Disk Drive Unit. Refer to the appropriate Assembly/Disassembly Procedure. Restore the system to its normal configuration and run the Live Box Procedure.

** If all the conditions detailed in Step J are verified, but the system fails CVFY Test 4 (see Table 3-2, Note #1), then go to Step K.

** If all the conditions detailed in Step J are verified, and the system still does not boot (load the Diagnostic Disk), then the DMA Floppy/Clock Module is bad. Replace it, and run the Live Box Procedure.

HALT
.SF3E0
F3E0 nnnn–300

Display at end of Step J, 17, c.

NOTE

Reinstall the regular PROM set. Refer to the appropriate Assembly/Disassembly procedure.

Step K. FLOPPY DISK DRIVE VERIFICATION (WRITE)

The purpose of this step is to diagnose those problems where, the Diagnostic Disk can be loaded but the CVFY test #4 fails, or where some other problem indicates that a Disk Unit can read information from the diskette but cannot write information correctly. It assumes that all the verifications called for in Step J can be made.

1. Load the CVFY program. Refer to the Live Box Procedure for details on CVFY usage.

2. Loop on test 4.

3. Set up the scope for:

Single channel	2V per cm.
Trig.	internal
Horz.	as necessary

4. Verification of the Write Enable signal from the Module.

a. Place the scope probe on U2 pin–11.

b. Verify that the correct waveform is present. This waveform should occur approximately every 4 seconds as the program initiates a write operation. It appears as a burst of rapid non-sychronous activity.

** If this condition is not met, replace the DMA-Floppy Clock Module. Refer to the appropriate Assembly/Disassembly Procedure. Restore the system to its normal configuration and run the Live Box Procedure.

5. Verification of the Write Data from the Module.

a. Move the scope probe from U2 pin-11 and place it on U2 pin-3. Verify that the write data is present.

b. Place the scope probe on U2 pin-3. Verify that the correct waveform is present. This waveform should occur approximately every 4 seconds as the program initiates a write operation. It appears as a burst of rapid non-sychronous activity.

** If this condition is not met, replace the DMA-Floppy Clock Module. Refer to the appropriate Assembly/Disassembly Procedure. Restore the system to its normal configuration and run the Live Box Procedure.

** If you have gone through Step K and found that the Module is sending the correct write signals, then the Floppy Disk Drive Unit is bad. Replace it, and run the Live Box Procedure.

STEP L. VIDEO PROBLEM (Unsynchronized display entry point)

The purpose of this step is to determine which submodule within the CRT Electronics section of the system is bad. You should be entering this step from Step D and should have already determined that the CRT Electronics are not operating properly.

NOTE

A display of a bright dot or line can burn the CRT face. Remove the jumper quickly if anything but a full screen display occurs.

** If you see a blank display, go to Step L, 1

** If you see a horizontal line, go to Step L, 3

** If you don't see any of the above, then call Fluke Service.

1. Blank Display--check filiaments

WARNING

HIGH VOLTAGE IS PRESENT WITHIN THE CRT AREA. DO NOT PLACE HANDS OR METALLIC OBJECTS INTO THIS AREA.

a. Look at the clear glass neck of the CRT, just ahead of the plastic tube socket.

b. Locate the filiament (glowing brightly).

** If you can see the filiament glowing and there is no blue glow from the neck area of the CRT, you can assume that the CRT is good and the CRT Electronics must be replaced. Refer to the appropriate Assembly/Disassembly Procedure. Following this, run the Live Box Procedure.

** If you can see a blue glow in the neck area of the CRT, the CRT is gassy and it must be replaced. Refer to the appropriate Assembly/Disassembly Procedure.

** If you cannot see the filiament glowing, then go on to Step L, 2.

2. Blank Display -- filiament not glowing

a. Turn system power off

WARNING

A POSSIBILITY EXISTS THAT THE HIGH VOLTAGE SUPPLY IS NOT DISCHARGED. FOLLOW THE PROCEDURE OUTLINED IN ASSEMBLY/DISASSEMBLY PROCEDURES (PARAGRAPH 3-37, 9) FOR DISCHARGING THE HIGH VOLTAGE SUPPLY BEFORE GOING ON.

b. After discharging the High Voltage Supply, remove the socket from the base of the CRT.

c. Measure between pins 1 and 8 to check filiament continuity. Five ohms is typical.

** If the filiaments are open, the CRT is bad and must be replaced. Refer to the appropriate Assembly/Disassembly procedures, especially the warning paragraph concerning the dangers involved with the process. Replace the CRT; restore the system to its normal configuration, and run the Live Box Procedure.

d. Measure the resistance in the horizontal yoke coil as follows:

1. Remove the red wire from the horizontal yoke coil.

2. Measure the resistance between the red and black tabs on the yoke coil.

3. The resistance should be 2.2 ohms, nominally. Some variation from this reading is acceptable; however, since there are two windings in parallel, a reading of 4.4 ohms indicates that one winding is open and an open reading indicates both windings are open. Neither of these conditions is acceptable.

** If the reading is not correct, then the yoke is defective. The yoke and CRT are provided as a matched, pre-aligned set for servicing; therefore, the whole CRT Assembly must be replaced. Refer to the appropriate Assembly/Disassembly procedure and replace the CRT Assembly. Carefully note the warning about the danger involved in handling the CRT. Following this, restore the system to its normal configuration.

3. Incorrect Display -- horizontal line
This condition is due to a fault in the vertical drive which can either be the yoke or the CRT electronics.

a. Remove the top cover.

WARNING

High voltage is present within the CRT area. Do not place your hands or metallic objects into this area.

b. Turn system power off.

c. Remove the yellow wire from the yoke. Measure the resistance between the yellow and green tabs on the yoke. There are two windings in series here. Therefore, an open reading is not acceptable. The correct reading is approximately 260 ohms.

** If the reading is not correct, the yoke is defective. The yoke and CRT are provided as a matched, pre-aligned set for servicing, therefore, the CRT must be replaced. Refer to the appropriate Assembly/Disassembly procedures, to replace the CRT assembly, then restore the system to its normal configuration.

** If the reading is correct then the CRT Electronics are faulty. Refer to the appropriate Assembly/Disassembly procedure; replace the CRT Electronics; restore the system to its normal configuration and run the Live Box Procedure.

STEP M. VIDEO ELECTRONICS PROBLEMS (Vertical roll entry point)

You should be entering this step from Step D. You have a display that has characters but displays them with a vertical roll. This problem is caused by a lack of vertical synchronization. The purpose of this step is to determine if the fault is in the Video/Keyboard Module or the CRT Electronics.

1. Remove the top cover from the instrument.

WARNING

High Voltage is present within the CRT area. Do not place your hands or metallic objects into this area.

2. Set up the scope as follows:

VERT single trace, 2V per cm.
SYNC internal, positive
HORZ as necessary

3. Place the scope probe on TP 9 on the Mother Board. You should be able to see positive-going TTL-level (4V approximate) pulses, approximately 190 us wide and 17 ms apart.

** If these pulses are present, then the CRT Electronics must be replaced. Refer to the appropriate Assembly/Disassembly procedure and replace the CRT Electronics. Restore the system to its normal configuration. Run the Live Box Procedure.

** If you do not see these pulses, disconnect the CRT Electronics from the system and look again. Proceed as follows:

1. Turn the power off.

2. Turn the instrument up on its side then remove the bottom cover.

3. Unplug the edge connector from the rear of the bottom CRT Electronics PCB.

4. Turn the power back on. The display is blank with the connector removed.

5. Using the scope, look at TP9 on the Mother Board.

** If the pulses are there, the CRT Electronics must be replaced. Refer to the appropriate Assembly/Dissassembly procedure and replace the CRT Electronics. Restore the system to its

normal configuration, and run the Live Box Procedure.

** If the pulses are not there, replace the Video/Keyboard Module. Refer to the appropriate Assembly/Disassembly procedure. Restore the system to its normal configuration, and run the Live Box Procedure.

STEP N. VIDEO ELECTRONICS PROBLEMS (Horizontal roll entry point)

You should be entering this step from Step D. The display has characters but displays them with a horizontal roll that shows up as a tearing in the horizontal direction. This problem is caused by a lack of horizontal synchronization. The purpose of this step is to find out if the problem is in the Video/Keyboard Module or in the CRT Electronics.

1. Remove the top cover from the instrument.

WARNING

HIGH VOLTAGE IS PRESENT WITHIN THE CRT AREA. DO NOT PUT YOUR HANDS OR METALLIC IMPLEMENTS INTO THIS AREA.

2. Set up the scope as follows:

VERT 2V per cm.
SYNC internal, positive
HORZ as necessary

3. Place the scope probe on TP-8 on the Mother Board. You should be able to see positive-going TTL pulses (approximately 4V) that are approximately 4.5 us wide and 62 us apart.

** If these pulses are present, replace the CRT Electronics. Refer to the appropriate Assembly/Disassembly Procedure. Restore the system its normal configuration, then run the Live Box Procedure.

** If you do not see these pulses, disconnect the CRT Electronics and look again. Proceed as follows:

1. Turn the power off.

2. Turn the instrument up on its side then remove the bottom cover.

3. Unplug the edge connector from the rear of the bottom CRT Electronics PCB.

4. Turn the power back on. The display is blank with the connector removed.

5. Using the scope, look at TP-9 on the Mother Board.

** If the pulses are there, replace the CRT Electronics. Refer to the appropriate Assembly/Disassembly procedure. Restore the system to its normal configuration and run the Live Box Procedure.

** If the pulses are not there, replace the Video/Keyboard Module. Refer to the appropriate Assembly/Disassembly procedure. Restore the system to its normal configuration, and run the Live Box Procedure.

STEP O. VIDEO DISPLAY PROBLEM (Bright display entry point)

You should be entering this step from Step D. The display is bright, covering the entire display area with light (without the jumper between TP5 and TP7 on the Mother Board). The problem can be caused by a fault in the CRT (shorted elements) or in various areas of the CRT Electronics that provide the correct bias for the CRT. The purpose of this step is to find out if the problem is in the CRT itself or in the CRT Electronics.

1. Make a note of the position of the Brightness Pot in the CRT Electronics section. Turn this pot CCW until the background raster just disappears.

** If the background raster will extinguish, return to the beginning of Step D.

** If the background raster will not extinguish, return the pot to its original position and go on to Step O,2.

2. Turn power off.

3. Remove the tube socket from the base of the CRT.

4. Set up the scope (or use a DVM) as follows:

VERT 10V per cm., DC coupled
SYNC Auto
HORZ 1 ms/cm

5. Turn system power back on.

6. Measure the voltage at the center tap of the Brightness Pot in the CRT Electronics. The voltage should be 5V ± 25V (i.e., -20V to +30V). If the

original setting is not certain, verify the adjustment range is at least –20V to +30V.

7. Measure the voltage at the yellow lead to the CRT socket at PCB end of the lead. This voltage should be approximately 70V (but not less than 60V) with no jumper between TP5 and TP7 on the Mother Board. It should drop to approximately 52V with the jumper.

** If the voltages called for in Step O, 6 and 7 are both correct, replace the CRT. If they are not both correct, replace the CRT Electronics. Refer to the appropriate Assembly/Disassembly procedure. Restore the system to its normal configuration, then run the Live Box Procedure.

3-20. ASSEMBLY/DISASSEMBLY PROCEDURES

3-21. Introduction

3-22. The following procedures allow the user to remove and replace any of the modules or assemblies mentioned in either of the diagnostic procedures found at the beginning of this section. The term "module" refers to one of the PCBs within the card cage. The term "assembly" denotes other replaceable portions of the 1720A. Each procedure has its own heading and appears in the table of contents.

3-23. Tools Required

3-24. The only tools required for most of the procedures that follow are:

1. 5-Inch Phillips Screwdriver
2. 5-Inch Slot Screwdriver
3. 24-Pin IC Extractor

3-25. In addition, if you are replacing the CRT Assembly or the Touch Sensitive Display, you need:

1. A 9-inch screwdriver (slot), with a 1/4 inch blade and a plastic handle at least 3 inches long.
2. 1 Kilohm, 1/2 watt, carbon resistor.
3. Two clip leads, with alligator clips, approximately 5 inches long.
4. Safety gloves.
5. Full face shield (preferred) or safety goggles.
6. Long sleeved jacket.
7. Work bench or table, large enough to accomodate the 1720A when laid flat and still leave enough room on either side to allow work on the Front Panel assembly.
8. Soft pad (foam or quilted) approximately 8 by 11 inches.
9. Plastic screwdriver, 9 inches long minimum, 1/4-inch tip approximate. It should be as rigid as possible to resist twisting during the adjustment.
10. Plastic hex-drive alignment tool (for pots in the CRT Electronics area).

3-26. Module (PCB)

3-27. All the PCBs within the 1720A card cage can be removed and replaced using the steps that follow. Refer to Table 3-2 or to the Interconnection Diagram in Section 6, for the slot/location of any module. Both the IEEE-488 Module and the E-Disk Module require one extra step that the others do not.

1. Turn power off and unplug the instrument.
2. Remove the rear card cage cover which is held by three captive retaining screws. Two of the screws are in the rear feet and the third screw is in the middle of the rear section.
3. If you are removing the IEEE-488 Module, remove the screws from the rear edge of the Module. They hold it into the card cage. Skip this step for all other Modules.
4. If you are removing the Electronic Disk Module, turn the E-Disk Battery switch to the DISABLE position. Skip this step for all other modules.
5. Use the PCB ejectors located on the rear edge of the Module to remove the PCB from its socket. To do this, place a thumb under each ejector and pull away from the PCB. As soon as the PCB is out of the socket, slide it all the way out of the card cage.
6. Slide the new PCB into the card cage until it comes to a stop against the socket. Press against the rear edge of the PCB with the palm of your hand to seat the PCB in the socket.
7. Replace the rear card-cage cover.

3-28. Power Supply

3-29. To remove and replace the Power Supply proceed as follows:

1. Turn the power off and unplug the instrument. Wait for at least three minutes before going on to the next step.

2. Back out the three captive retaining screws that hold the Power Supply in. Two of these screws are located in the rear feet and the third screw is located in the center of the rear panel.

3. Slide the Power Supply out of the instrument.

4. Slide the new Power Supply into the unit.

5. Tighten the three captive retaining screws.

6. Refer to the Dead Box Procedure, Step B, and verify that the new supply operates correctly.

3-30. Battery

3-31. To remove and replace the battery, proceed as follows:

1. Turn the power off and unplug the instrument. Wait for at least three minutes before going on to the next step.

2. Back out the three captive retaining screws that hold the Power Supply in. Two of these screws are located in the rear feet and the third screw is located in the center of the rear panel.

3. Slide the Power Supply out of the instrument.

4. Place the Power Supply top down on the bench.

5. Remove the four screws that hold the bottom Power Supply PCB to the main assembly.

6. Gently separate the PCB from the main assembly and lay it to one side at the limit of the wires.

7. Unplug the battery by pulling on the battery plug where it enters the jack at the PCB.

8. Push up on the bottom of the battery to remove it from the battery holder. The battery wires and plug will pass through the bottom of the holder.

9. Install a new battery, routing the wires and plug through the bottom of the holder.

10. Press the battery plug into the jack at the PCB.

11. Replace the PCB on the main assembly.

12. Install the four screws that hold the bottom Power Supply PCB to the main assembly.

13. Slide the Power Supply into the instrument.

14. Tighten the three captive retaining screws that hold the Power Supply in. Two of these screws are located in the rear feet and the third screw is located in the center of the rear panel.

3-32. Floppy Disk Drive

3-33. To remove and replace the Floppy Disk Drive Unit, proceed as follows:

1. Turn power off and unplug the instrument.

2. Remove the top and bottom covers from the instrument.

3. Remove the CPU Module using the Module (PCB) Assembly/Disassembly Procedure.

4. Disconnect the Floppy Drive Unit power cable.

5. Turn the instrument up on one side.

6. Hold the Floppy Disk Drive Unit with one hand while removing the four screws that pass through the bottom mounting plate into the Floppy Disk Drive Unit cover. Be careful not to drop the Floppy Disk Drive when you remove the last screw.

7. Lay the instrument bottom side down, maintaining your hold on the Floppy Disk Drive Unit.

8. Slide the Drive Unit back until it clears the Front Panel. Lift the unit upwards until you can disconnect the ribbon cable. Notice that the red stripe on the edge of the ribbon cable goes towards the center of the 1720A. Since the connector is not keyed, this red stripe provides the only means of correctly reconnecting the cable.

9. Remove the four screws holding the sheet metal cover to the Floppy Disk Drive. Transfer the cover to the new drive and replace the four screws.

10. Connect the ribbon cable to the new Drive, keeping the red stripe towards the center of the instrument. Slide the Drive Unit into the instrument.

11. Hold the Floppy Disk Unit in place and tilt the instrument up on one side. Replace the four screws that pass through the bottom of the Drive Unit Cover. Don't let go of the Drive Unit until you have installed at least one of the screws.

12. Reconnect the power cable.

13. Install the CPU Module.

14. Install the top and bottom covers.

3-34. CRT Assembly

3-35. To remove and replace the CRT Assembly, proceed as follows:

WARNING

BE CAREFUL WHEN YOU HANDLE THE CRT. WEAR PROTECTIVE CLOTHING AND SAFETY GLASSES OR A FULL FACE SHIELD. AVOID STRIKING THE CRT ON ANY OBJECT WHICH MIGHT CAUSE IT TO CRACK OR IMPLODE.

1. Turn the power off and unplug the instrument.

2. Remove the rear card-cage cover which is held by three captive retaining screws. Two of the screws are in the rear feet and the third screw is in the middle of the rear section.

3. Remove the five screws on the rear panel that hold the IEEE-488 Module in place and remove the module.

4. Tilt the 1720A up on its side and remove the top and bottom covers from the instrument.

5. Disconnect the ribbon cable, that connects the Front Panel to the instrument, from the Front Panel. Notice that the red stripe on the edge of the cable goes towards the side of the instrument. Since this connector is not keyed, this red stripe provides the only way to reconnect the cable correctly.

6. Disconnect the multi-color ribbon cable from the Mother Board. Free the cable from the clamp on the Floppy Drive mounting plate. Note that the brown wire on the edge of the cable goes toward the center of the instrument. Since this connector is not keyed, the position of this brown wire provides the only means to reconnect the cable correctly.

7. Remove the two screws that pass through the Floppy Disk Drive Unit support bracket into the Front Panel bezel.

8. Turn the box bottom side down. Do not crush the loose ribbon cables or connectors against the bench.

WARNING

BOTH THE CRT ANODE AND THE HIGH VOLTAGE SUPPLY MAY RETAIN A HIGH VOLTAGE CHARGE AFTER THE INSTRUMENT HAS BEEN TURNED OFF FOR SOME TIME. IN THE CASE OF THE CRT ANODE THIS CHARGE CAN BUILD BACK UP EVEN AFTER BEING DISCHARGED BY THE PROCEDURE THAT FOLLOWS. ALTHOUGH NOT CONTAINING ENOUGH ENERGY TO BE HARMFUL, THE CHARGE CAN DELIVER A SHOCK THAT COULD CAUSE THE CRT TO BE DROPPED.

9. Discharge the high voltage charge through a 1 Kilohm resistor as follows:

a. Connect one end of the resistor to the chassis with one clip lead.

b. Connect the other end of the resistor to the shaft of the 9 inch screwdriver with the 1/4 inch tip, using the other clip lead.

c. Hold the screwdriver by its plastic handle and gently slip it under the edge of the plastic nipple on the CRT end of the high voltage lead; keep the blade flat against the glass envelope. Slide the blade forward until the screwdriver blade touches the metallic clip at the end of the high voltage lead. You must be careful not to scratch the surface of the CRT.

10. Remove the High Voltage lead from the side of the CRT as follows:

a. Grasp the rubber insulating nipple firmly and slide it sideways against the tension of the wire prongs that are inside the nipple. This should unlock the connector.

b. Pull the nipple outwards while maintaining the sideways pressure to remove the connector.

c. Notice how the High Voltage cable is dressed. It is important to copy this dressing when reconnecting the High Voltage lead because it minimizes corona losses by staying as far from all grounded surfaces as possible.

11. Remove the rear connector from the base end of the CRT neck.

12. Remove the 4 wires from the yoke. Note that they are on color coded tabs.

13. Remove the side decals from the handles. New decals are provided in the kit for the CRT Assembly.

14. Remove the four screws that hold each handle to the side of the instrument.

15. Ease the Front Panel forward, away from the instrument, until it is just free from the sides.

16. Gently pull the Front Panel toward you until the neck of the CRT is free of the instrument. Notice how the Front Panel ribbon cable is dressed, particularly where it passes the CRT support bracket. To avoid damage to the cable, duplicate this dressing when you reassemble the instrument.

17. Set the Front Panel face down on a clear area of the bench with the neck of the CRT pointing up.

18. Don protective clothing:

a. Full face shield (preferred) or protective goggles.

b. Long sleeved jacket.

c. Protective gloves.

19. Remove the four screws from the CRT support brackets.

20. Remove the support brackets.

21. Grasp the CRT on each side of the bulb. Lift it straight up being careful to leave the face pointing down. Do not strike it against anything.

22. Gently set the CRT face-down on a soft pad in a clear protected area of the bench. The purpose of this pad is to prevent scratching the face of the CRT and to avoid placing the CRT on a hard surface. If theTouch-Sensitive Display is stuck to the CRT, remove it and place it back into the Front Panel, but keep the CRT face down at all times.

23. Unpack the new CRT Assembly, using your protective clothing and being careful not to strike the CRT against anything that may crack the glass envelope.

a. Remove the protective cover from the base of the CRT.

b. Hold the CRT by the bulb, curling your fingers around the edges onto the face.

c. Place it face down, into the Front Panel with the socket for the High Voltage Lead towards the center of the panel.

24. Install the CRT support brackets and the four screws that secure the CRT to the Front Panel. You may remove your protective clothing at this time.

25. Observe that each side of the front panel has guides that are meant to accomodate notches that are in the sheet metal sides of the instrument. Carefully rotate the Front Panel away from you to place it in the correct relative position that allows you to reinstall it in the instrument. The guides on each side of the Front Panel should now be directly in line with the notches in the sheet metal sides.

26. Route the Front Panel ribbon cable in such a way that it will not be sheared off by the sheet metal portion of the box on the underside of the instrument when you replace the Front Panel.

27. Move the Front Panel toward the instrument, being careful that the sheet metal notches correctly enter the guides on each side of the Front Panel. Continue this movement until the Front Panel is all the way onto the instrument and the guides are all the way into the notches. The holes in the handles should match up with the corresponding holes in the Front Panel. Insure that the ground spring is making proper contact with the outside of the CRT.

28. Install the four screws on each handle that hold it to the side of the instrument.

29. Replace the side decals on the handles. New decals are provided in the kit for the CRT Assembly.

30. Connect the 4 wires to the yoke. Note that they are on color coded tabs.

31. Attach the rear connector to the socket end of the CRT neck.

32. Connect the High Voltage lead on the side of the CRT as follows:

a. Grasp the rubber insulating nipple firmly and press the anode connector into the socket on the side of the CRT.

b. It is important to copy the original dressing when reconnecting the High Voltage lead because it minimizes corona losses by staying as far from all grounded surfaces as possible.

33. Turn the box top side up. Do not crush the loose ribbon cables or connectors against the bench.

34. Install the two screws that pass through the Floppy Disk Drive Unit support bracket into the Front Panel bezel.

35. Connect the multi-color ribbon cable from the Mother Board. Insert the cable into the clamp on the Floppy Drive mounting plate. Note that the brown wire on the edge of the cable goes toward the center of the instrument. Since this connector is not keyed, the position of this brown wire provides the only means to reconnect the cable correctly.

36. Connect the ribbon cable to the Front Panel. Notice that the red stripe on the edge of the cable goes toward the side of the instrument. Since this connector is not keyed, this red stripe provides the only way to reconnect the cable correctly.

37. Tilt the 1720A up on its side and replace the top and bottom covers on the instrument.

38. Install the IEEE Module, insuring that it is firmly seated. Install five screws that hold the IEEE Module in place.

39. Install the rear card cage cover which is held by three captive retaining screws. Two of the screws are in the rear feet and the third screw is in the middle of the rear section.

40. Plug the instrument in and go through the Video Alignment Procedure given in paragraph 3-50.

3-36. Touch-Sensitive Display

3-37. To replace the Touch-Sensitive Display, proceed as follows:

WARNING

BE CAREFUL WHEN YOU HANDLE THE CRT. WEAR PROTECTIVE CLOTHING AND SAFETY GLASSES OR A FULL FACE SHIELD. AVOID STRIKING THE CRT ON ANY OBJECT WHICH MIGHT CAUSE IT TO CRACK OR IMPLODE.

1. Turn the power off and unplug the instrument.

2. Remove the rear card-cage cover which is held by three captive retaining screws. Two of the screws are in the rear feet and the third screw is in the middle of the rear section.

3. Remove the five screws on the rear panel that hold the IEEE-488 Module in place and remove the module.

4. Tilt the 1720A up on its side, and remove the top and bottom covers from the instrument.

5. Disconnect the ribbon cable, that connects the Front Panel to the instrument, from the Front Panel. Notice that the red stripe on the edge of the cable goes towards the side of the instrument. Since this connector is not keyed, this red stripe provides the only way to reconnect the cable correctly.

6. Disconnect the multi-color ribbon cable from the Mother Board. Free the cable from the clamp on the Floppy Drive mounting plate. Note that the brown wire on the edge of the cable goes toward the center of the instrument. Since this connector is not keyed, the position of this brown wire provides the only means to reconnect the cable correctly.

7. Remove the two screws that pass through the Floppy Disk Drive Unit support bracket into the Front Panel bezel.

8. Turn the box bottom side down. Do not crush the loose ribbon cables or connectors against the bench.

WARNING

BOTH THE CRT ANODE AND THE HIGH VOLTAGE SUPPLY MAY RETAIN A HIGH VOLTAGE CHARGE AFTER THE INSTRUMENT HAS BEEN TURNED OFF. DO NOT SKIP THE FOLLOWING STEP WHICH SAFELY REMOVES THIS CHARGE.

9. Discharge the high voltage charge through a 1 kilohm resistor as follows:

a. Connect one end of the resistor to the chassis with one clip lead.

b. Using the other clip lead, connect the other end of the resistor to the shaft of the 9-inch screwdriver with the 1/4-tip.

c. Hold the screwdriver by its plastic handle and gently slip it under the edge of the plastic nipple on the CRT end of the high voltage lead. Keep the blade flat against the glass envelope. Slide the screwdriver blade forward until it touches the metallic clip at the end of the high voltage lead. Be careful not to scratch the surface of the CRT.

10. Remove the High Voltage lead from the side of the CRT as follows:

a. Grasp the rubber insulating nipple firmly and slide it sideways against the tension of the wire prongs that are inside the nipple. This should unlock the connector.

b. Pull the nipple outwards while maintaining the sideways pressure to remove the connector.

c. Notice how the High Voltage cable is dressed. It is important to copy this dressing when reconnecting the High Voltage lead because this minimizes corona losses by staying as far from all grounded surfaces as possible.

11. Remove the rear connector from the base of the CRT neck.

12. Remove the 4 wires from the yoke. Note that they are on color coded tabs.

13. Remove the side decals from the handles. New decals are provided in the kit for the CRT Assembly.

14. Remove the four screws that hold each handle to the side of the instrument.

15. Ease the Front Panel forward, away from the instrument, until it is just free from the sides.

16. Gently pull the Front Panel towards you until the neck of the CRT is free of the instrument. Notice how the Front Panel ribbon cable is dressed, particularly where it passes the CRT support bracket. To avoid damage to the cable, duplicate this dressing when you reassemble the instrument.

17. Set the Front Panel face down on a clear area of the bench with the neck of the CRT pointing upwards.

18. Don protective clothing:

a. Full face shield (preferred) or protective goggles.

b. Long sleeved jacket.

c. Protective gloves.

19. Remove the four screws from the CRT support brackets.

20. Remove the support brackets.

21. Grasp the CRT on each side of the bulb. Lift it straight up being careful to leave the face pointing down. Do not strike it against anything.

22. Gently set the CRT face-down on a soft pad in a clear, protected area of the bench. If the Touch-Sensitive Display is stuck to the CRT, remove it and place it to one side. Note the routing of the Display cable. Keep the CRT face down at all times.

23. Lift out the old Touch-Sensitive Display (if it is still in the Front Panel), taking care to note the routing of the ribbon cable. Put the new Touch-Sensitive Display in, using the same cable routing.

24. Install the CRT Assembly, using your protective clothing. Be careful not to strike the CRT against anything that may crack the glass envelope.

a. Hold the CRT by the bulb, curling your fingers around the edges onto the face.

b. Place it face down, into the Front Panel with the socket for the High Voltage Lead towards the center of the panel.

25. Install the CRT Support brackets and the four screws that secure the CRT to the Front Panel. You may remove your protective clothing at this time.

26. Observe that each side of the front panel has guides that are meant to accomodate notches that are in the sheet metal sides of the instrument. Carefully rotate the Front Panel away from you to place it in the correct relative position that will allow you to reinstall it in the instrument. The guides on each side of the Front Panel should now be directly in line with the notches in the sheet metal sides.

27. Route the Front Panel ribbon cable in such a way that it will not be sheared off by the sheet metal portion of the box on the underside of the instrument when you replace the Front Panel.

28. Move the Front Panel towards the instrument, being careful that the sheet metal notches correctly enter the guides on each side of the Front Panel. Continue this movement until the Front Panel is all the way onto the instrument and the guides are all the way into the notches. The holes in the handles should match up with the corresponding holes in the Front Panel. Insure that the ground spring is making proper contact with the ouside of the CRT.

29. Install the four screws from each handle that hold it to the side of the instrument.

30. Replace the side decals from the handles. New decals are provided in the kit for the CRT Assembly.

31. Connect the 4 wires to the yoke. Note that they are on color coded tabs.

32. Connect the rear connector to the base of the CRT neck.

33. Connect the High Voltage lead to the side of the CRT as follows:

a. Grasp the rubber insulating nipple firmly and press the anode connector into the socket on the side of the CRT.

b. It is important to copy the original dressing when reconnecting the High Voltage lead because it minimizes corona losses by staying as far from all grounded surfaces as possible.

34. Turn the box bottom side down. Do not crush the loose ribbon cables or connectors against the bench.

35. Install the two screws that pass through the Floppy Disk Drive Unit support bracket into the Front Panel bezel.

36. Connect the multi-color ribbon cable from the Mother Board. Insert the cable into the clamp on the Floppy Drive mounting plate. Note that the brown wire on the edge of the cable goes towards the center of the instrument. Since this connector is not keyed, the position of this brown wire provides the only means to reconnect the cable correctly.

37. Connect the ribbon cable at the Front Panel. Notice that the red stripe on the edge of the cable goes towards the side of the instrument. Since this connector is not keyed, this red stripe provides the only way to reconnect the cable correctly.

38. Tilt the 1720A up on its side, and install the top and bottom covers from the instrument.

39. Install the IEEE Module, insuring that it is firmly seated. Install the five screws that hold the IEEE Module in place.

40. Install the rear card-cage cover which is held by three captive retaining screws. Two of the screws are in the rear feet, and the third screw is in the middle of the rear section.

3-38. CRT Electronics Assembly/Disassembly Procedure

3-39. REPLACEMENT OF THE HIGH VOLTAGE MODULE

1. Turn the power off, and unplug the instrument.

2. Remove the top and bottom covers from the instrument.

WARNING

BOTH THE CRT ANODE AND THE HIGH VOLTAGE SUPPLY MAY RETAIN A HIGH VOLTAGE CHARGE AFTER THE INSTRUMENT HAS BEEN TURNED OFF FOR SOME TIME. DO NOT SKIP THE FOLLOWING STEP WHICH SAFELY REMOVES THIS CHARGE.

3. Discharge the high voltage charge through a 1 Kilohm resistor as follows:

a. Connect one end of the resistor to the chassis with one clip lead.

b. Connect the other end of the resistor to the shaft of the 9-inch screwdriver with the 1/4-inch tip using the other clip lead.

c. Hold the screwdriver by its plastic handle and gently slip it under the edge of the plastic nipple on the CRT end of the high voltage lead. Keep the blade flat against the glass envelope. Slide the screwdriver blade forward until it touches the metallic clip at the end of the high voltagelead. You must be careful not to scratch the surface of the CRT.

4. Remove the High Voltage lead from the side of the CRT as follows:

a. Grasp the rubber insulating nipple firmly and slide it sideways against the tension of the wire prongs that are inside the nipple. This should unlock the connector.

b. Pull the nipple outwards while maintaining the sideways pressure to remove the connector.

c. Notice how the High Voltage cable is dressed. It is important to copy this dressing when reconnecting the High Voltage lead because it minimizes corona losses by staying as far from all grounded surfaces as possible.

5. Remove the three flyback wires from the CRT Electronics PCB at the side of the unit. Notice that the wire color codes are labeled on the PCB.

6. Remove the two wires from the High Voltage Module. The orientation of these two wires is not critical.

7. Remove the ground wire by removing the screw that passes through the tab into the High Voltage Module.

8. Tilt the instrument up on one side, bottom towards you.

9. Remove the four screws from the bottom that hold the High Voltage Module in place. To prevent the High Voltage Module from falling onto the CRT, support the module with one hand as you remove the last screw.

10. Remove the High Voltage Module.

11. Remove the sheet metal cover from the High Voltage Module. Observe the lead dress of the HV transformer lead to the diode. This dress must be duplicated upon reassembly to minimize corona.

12. Pull the diode out of the white insulator connector, leaving it in the lead to the HV transformer.

13. Remove the three screws that hold the high voltage transformer in place.

14. Remove the High Voltage Transformer.

15. Replace the High Voltage Transformer and the diode that comes with it. Install the three screws that hold it in place.

16. Push the new diode into the white insulator connector.

NOTE

The lead dress of the HV wire to the diode inside the High Voltage Module is critical. Dress it in an "s" loop over the transformer winding and away from the cover and core.

17. Install the sheet metal cover.

18. Install the two screws that hold the insulator in the sheet metal cover.

19. Place the High Voltage Assembly in position and hold it there as you tilt the instrument up on its side (with the bottom of the instrument toward you).

20. Install the four screws that hold the assembly in place. Support the assembly until you have installed at least one of the screws.

21. Attach the ground wire to the screw that passed through the tab into the High Voltage Module.

22. Replace the two wires to the High Voltage Module. The orientation of these two wires is not critical.

23. Replace the three flyback wires to the CRT Electronics PCB at the side of the unit. Notice that the wire color codes are labeled on the PCB.

24. Connect the high voltage lead by pressing the two prongs inside the nipple into the socket on the side of the CRT Assembly. Be sure to duplicate the dressing of the original lead. This minimizes corona losses by keeping the leads as far from all grounded surfaces as possible.

3-40. REPLACEMENT OF THE CRT ELECTRONICS PCB

1. Unplug the four yoke wires.

2. Remove the three flyback wires from the CRT Electronics PCB at the side of the unit. Notice that the wire color codes are labeled on the PCB.

3. Remove the socket from the base of the CRT.

4. Remove the Raster Pot from the chassis bracket.

5. Remove the four screws from the side of the instrument that hold the CRT Electronics Assembly.

6. Tilt the instrument up onto the side opposite to the CRT. This allows the CRT Electronics Assembly to rest on the bench.

7. Position the new assembly on the bench in place of the old.

8. Lay the instrument back down over the assembly.

9. Install the four screws (in the side of the instrument) that hold the CRT Electronics Assembly.

10. Install the new Raster Pot.

11. Connect the socket on the base of the CRT.

12. Connect the three flyback wires on the CRT Electronics PCB at the side of the unit. Notice that the wire color codes are labeled on the PCB.

13. Connect the four yoke wires, matching colors on the wires to the colors on the tabs.

14. Install the top and bottom covers.

15. Plug the instrument in and go through the Video Monitor Alignment Procedure.

3-41. Battery Fuse

3-42. To remove and replace the battery fuse, proceed as follows:

1. Turn the power off, and unplug the instrument. Wait for at least three minutes before going on to the next step.

2. Back out the three captive retaining screws that hold the Power Supply in. Two of these screws are located in the rear feet and the third screw is located in the center of the rear panel.

3. Slide the Power Supply out of the instrument.

4. Place the Power Supply top down on the bench.

5. Remove the four screws that hold the bottom Power Supply PCB to the main assembly.

6. Gently separate the PCB from the main assembly and lay it to one side at the limit of the wires.

7. Locate F1 on the PCB and remove it.

8. Replace F1 with a 5A Slo-blo fuse.

9. Position the Power Supply PCB on the main assembly.

10. Replace the four screws that hold the bottom Power Supply PCB to the main assembly.

11. Slide the Power Supply into the instrument.

12. Tighten the three captive retaining screws that hold the Power Supply In. Two of these screws are located in the rear feet and the third screw is located in the center of the rear panel.

3-43. HDT and Standard Prom Sets

3-44. HDT PROM Set

3-45. To remove the standard PROM Set and install the HDT PROM Set proceed as follows:

CAUTION

Static discharge can damage MOS components. Observe the cautions outlined at the beginning of this section in the STATIC AWARENESS notice while removing or replacing PROMs.

1. Turn the power off.

2. Locate the two PROMs.

3. Use a 24–Pin IC extractor to remove the PROMs. Store the removed PROMs in an appropriate anti-static container.

4. Notice that each of the HDT PROMs has a different part number. Locate the correct socket for each PROM.

5. Insert the HDT PROMs into the sockets with care, making sure that:

   a. The PROMs are orientated correctly with the locating key on the IC package in the same position as shown.

   b. No IC pin is bent or outside its hole.

3-46. Standard PROM Set

3-47. To remove the HDT PROM Set and install the Standard PROM Set, proceed as follows:

CAUTION

Static discharge can damage MOS components. Observe the precautions outlined at the beginning of this section in the STATIC AWARENESS notice while removing or replacing PROMs.

1. Turn the power off and unplug the instrument.

2. Locate the two PROMs.

3. Use a 24-Pin IC extractor to remove the PROMs. Store the removed PROMs in an appropriate anti-static container.

4. Notice that each of the standard PROMs has a different part number. Locate the correct socket for each PROM.

5. Insert the standard PROMs into the sockets with care, making sure that:

a. The PROMs are orientated correctly with the locating key on the IC package in the same position as shown.

b. No IC pin is bent or outside its hole.

3-48. Programmer Keyboard PCB

3-49. To remove and replace the Programmer Keyboard PCB proceed as follows:

1. Unplug the Programmer Keyboard from the instrument and lay it upside down on the bench.

2. Remove the four screws (from the bottom of the unit) that hold the two halves of the case together.

3. Lift the bottom off to the extent of the cable.

4. Pull the PCB up and out, unplugging the cable from the edge connector on the PCB.

5. Plug the new PCB in, using the cable you just unplugged from the old one.

6. Install the new PCB in the top half of the case (which is face down on the bench), insuring that the two alignment pegs on the case go into the holes in the PCB.

7. Push the cable into the U shaped channel next to the connector and press it into the groove in the side of the support post that sticks up to meet the bottom plate of the case.

8. Install the bottom plate. The cable should exit as before.

9. Install the four screws that hold the two halves of the case together.

3-50. Video Display Alignment Procedure

3-51. This procedure can be used after replacement of either the CRT Electronics or the CRT Assembly or used as reference material at any time the display needs adjustment.

NOTE

Display alignment is affected by temperature, supply voltage, brightness, and external magnetic fields. These effects should be considered before any adjustments are made.

3-52. To align the display, proceed as follows:

1. Turn the power on and allow the instrument to warm up to normal operating temperature.

NOTE

Changes in temperature will typically affect vertical size and position by one-half of a character and affect horizontal size and position by two characters.

2. The 70V supply must be 70V ± 0.5V. Use the 5V adjustment to bring it into tolerance if necessary, but make sure that this adjustment does not force the other supplies out of tolerance. Refer to the Power Supply Verification topic in the Dead Box Procedure if you have problems.

3. Establish the CRT alignment display by running the VIDEO program:

a. Press CTRL P on the Programmer keyboard to get into the CONSOLE MONITOR and get the "#" prompt character.

b. Enter VIDEO (CR).

c. Wait for the CRT alignment pattern to appear.

d. To check the alignment, use the transparent CRT Alignment Guage supplied in the Customer Maintenance Kit or purchased separately. Figure 3-5 shows this guage as it would appear when placed over the pattern. At normal operating temperatures (25 C ± 5 C), the vertical size and position should be within one-half of a character and the horizontal size and position should be within one character at the reference marks on the guage.

4. Touch-up the intensity by using the Brightness Pot on the rear of the Video Module PCB. The final brightness level should suit the ambient lighting level of the area the instrument will be used in without causing the characters to bloom.

5. Adjust the Raster Pot in the CRT Electronics area.

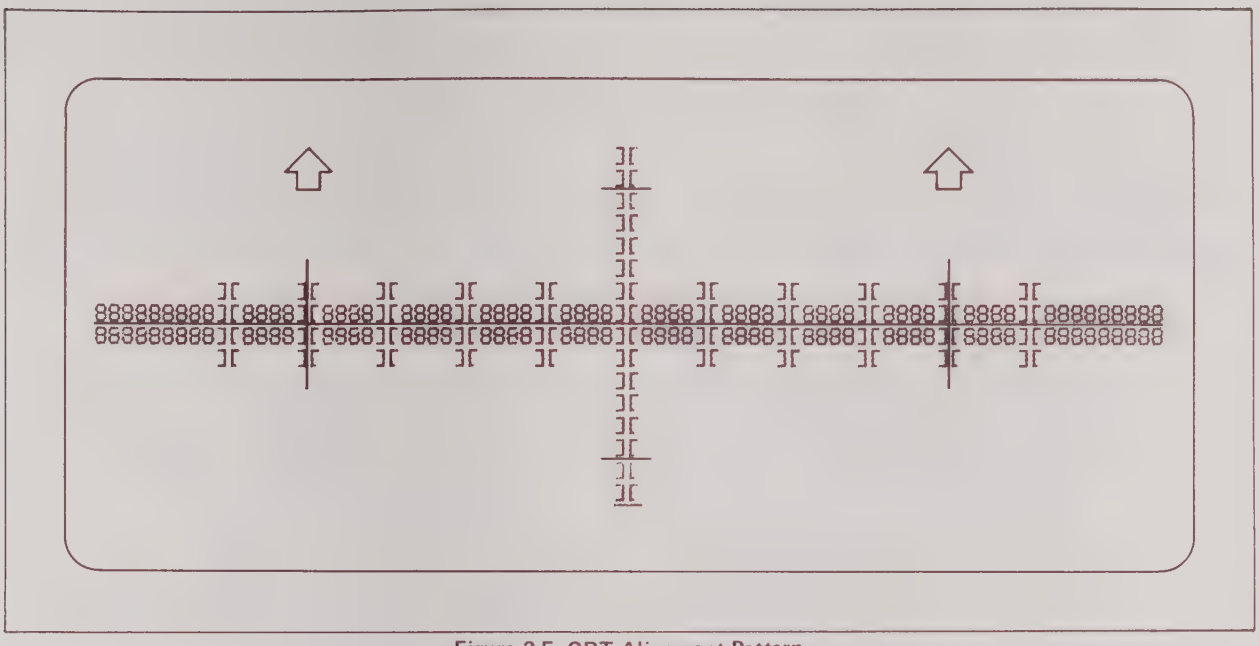


Figure 3-5. CRT Alignment Pattern

NOTE

This pot is factory adjusted to set maximum brightness and should not be adjusted unless the background raster becomes visible during normal usage or new assemblies (CRT Assembly or CRT Electronics) have been installed.

a. Turn the Raster Pot CW until the background raster becomes visible. If you adjust the pot all the way CW and the background raster does not appear, the Master Brightness Pot in the CRT Electronics area must be adjusted. Use a plastic screwdriver on this pot to avoid accidently shorting out adjacent components. Adjust as follows:

1. Turn the Raster Pot in the CRT Electronics area back approximately 1/4 turn.

2. Advance the Master Brightness Pot CW until the background raster just appears.

3. Turn the Raster Pot in the CRT Electronics area CCW until the background raster just disappears. If this cannot be done, the Master Brightness Pot is set too far CW.

b. Back off the adjustment (CCW) until the raster just disappears.

6. Do this step only if new CRT Electronics have been installed. Verify that the video display is centered in the raster as follows:

a. Turn the Raster Pot CW until the raster is visible.

b. Adjust the Horizontal Size coil L53 and the yoke centering rings as necessary so that both right and left sides of the raster can be seen.

NOTE

The centering rings should be adjusted only if really necessary. Notice the present position of the rings and put little tabs of masking tape on the yoke to mark the position they are in.

1. When you move the centering rings together, the raster moves in a circle. If you move the rings individually, the circle of movement changes diameter. Try to center the raster by moving the rings together. Move them individually only if the range of adjustment is not correct.

2. If all movements make things worse, return the rings to their original position and start over.

c. Adjust the Horizontal Osc Coil L50 to center the video display in the center of the raster.

d. If the display is not centered vertically, replace the Video Board.

e. Reduce the raster to the cut off point.

7. Adjust the yoke tilt as follows:

a. Roughly center the display using the centering rings.

1. When you move these rings together, the display moves in a circle. If you move the rings individually, the circle of movement changes diameter. Try to center the display by moving the rings together. Move them individually only if the range of adjustment is not correct.

2. If all movements make things worse, return the rings to their original position and start over.

b. Roughly set the horizontal size and vertical linearity using the Horizontal Size and Vertical Linearity pots.

c. Loosen the yoke clamp screw.

d. Position the alignment guage on the display. Rotate the yoke until the center rows of characters displayed are aligned with the center line of the guage.

e. Insure that the yoke is positioned tightly against the curved funnel of the CRT and tighten the yoke clamp screw to 6-inch pounds.

f. Verify that the tilt is still in adjustment and readjust as necessary.

8. Display Alignment

a. Adjust the horizontal size and centering for alignment to the two vertical marks under the arrows on the guage to within one-half of a character. Use the Horizontal Size Pot and the Centering rings.

b. Adjust the vertical size, centering and linearity so that the center column of characters is aligned with the horizontal marks within one-half of a character. Use the Vertical Size Pot, the Vertical Linearity Pot, and the Centering Rings as necessary.

9. Focus

CAUTION

Use an insulated screw driver.

a. Locate the Focus Pot R70.

b. Adjust for overall focus.

3-53. PREVENTATIVE MAINTENANCE PROGRAM

3-54. The Preventative Maintenance Program for the 1720A consists of three parts.

Part 1

Run the Live Box Procedure every 90 days. This insures that the system is operating correctly.

Part 2

Clean the Air Filter on the rear of the instrument every 90 days or as site conditions dictate. Instructions for cleaning the Air Filter are printed on the rear of the instrument.

Part 3

Clean the Touch-Sensitive Display every 90 days or as site conditions dictate. Use a moist, soft (non-abrasive) cloth and a mild soap solution to clean the Display.

Section 4
General Information

4-1. This section of the manual contains:

List of Abbreviations

Federal Supply Codes for Manufactures

Fluke Technical Service Centers - U.S. and Canada

Fluke Technical Service Centers - International

Sales Representatives - U.S. and Canada

Sales Representatives - International

List of Abbreviations and Symbols

A or amp	ampere
ac	alternating current
af	audio frequency
a/d	analog-to-digital
assy	assembly
AWG	american wire gauge
B	bel
bcd	binary coded decimal
°C	Celsius
cap	capacitor
ccw	counterclockwise
cer	ceramic
cermet	ceramic to metal(seal)
ckt	circuit
cm	centimeter
cmrr	common mode rejection ratio
comp	composition
cont	continue
crt	cathode-ray tube
cw	clockwise
d/a	digital-to-analog
dac	digital-to-analog converter
dB	decibel
dc	direct current
dmm	digital multimeter
dvm	digital voltmeter
elect	electrolytic
ext	external
F	farad
°F	Fahrenheit
FET	Field-effect transistor
ff	flip-flop
freq	frequency
FSN	federal stock number
g	gram
G	giga (10^9)
gd	guard
Ge	germanium
GHz	gigahertz
gmv	guaranteed minimum value
gnd	ground
H	henry
hd	heavy duty
hf	high frequency
Hz	hertz
IC	integrated circuit
if	intermediate frequency
in	inch(es)
intl	internal
I/O	input/output
k	kilo (10^3)
kHz	kilohertz
kΩ	kilohm(s)
kV	kilovolt(s)
lf	low frequency
LED	light-emitting diode
LSB	least significant bit
LSD	least significant digit
M	mega (10^6)
m	milli (10^{-3})
mA	milliampere(s)
max	maximum
mf	metal film
MHz	megahertz
min	minimum
mm	millimeter
ms	millisecond
MSB	most significant bit
MSD	most significant digit
MTBF	mean time between failures
MTTR	mean time to repair
mV	millivolt(s)
mv	multivibrator
MΩ	megohm(s)
n	nano (10^{-9})
na	not applicable
NC	normally closed
(-) or neg	negative
NO	normally open
ns	nanosecond
opnl ampl	operational amplifier
p	pico (10^{-12})
para	paragraph
pcb	printed circuit board
pF	picofarad
pn	part number
(+) or pos	positive
pot	potentiometer
p-p	peak-to-peak
ppm	parts per million
PROM	programmablle read-only memory
psi	pound-force per square inch
RAM	random-access memory
rf	radio frequency
rms	root mean square
ROM	read-only memory
s or sec	second (time)
scope	oscilloscope
SH	shield
Si	silicon
serno	serial number
sr	shift register
Ta	tantalum
tb	terminal board
tc	temperature coefficient or temperature compensating
tcxo	temperature compensated crystal oscillator
tp	test point
u or μ	micro (10^{-6})
uhf	ultra high frequency
us or μs	microsecond(s) (10^{-6})
uut	unit under test
V	volt
v	voltage
var	variable
vco	voltage controlled oscillator
vhf	very high frequency
vlf	very low frequency
W	watt(s)
ww	wire wound
xfmr	transformer
xstr	transistor
xtal	crystal
xtlo	crystal oscillator
Ω	ohm(s)
μ	micro (10^{-6})

Federal Supply Codes for Manufacturers

00213
Nytronics Comp. Group Inc.
Subsidiary of Nytronics Inc.
Formerly Sage Electronics
Rochester, New York

00327
Welwyn International, Inc.
Westlake, Ohio

00656
Aerovox Corp.
New Bedford, Massachusetts

00686
Film Capacitors, Inc.
Passaic, New Jersey

00779
AMP Inc.
Harrisburg, Pennsylvania

01121
Allen-Bradley Co.
Milwaukee, Wisconsin

01281
TRW Electronic Comp.
Semiconductor Operations
Lawndale, California

01295
Texas Instruments, Inc.
Semiconductor Group
Dallas, Texas

01537
Motorola Communications &
Electronics Inc.
Franklin Park, Illinois

01686
RCL Electronics Inc.
Manchester, New Hampshire

01730
Replaced by 73586

01884
Use 56289
Sprague Electric Co.
Dearborn Electronic Div.
Lockwood, Florida

02114
Ferroxcube Corp.
Saugerties, New York

02131
General Instrument Corp.
Harris ASW Div.
Westwood, Maine

02395
Rason Mfg. Co.
Brooklyn, New York

02533
Snelgrove, C.R. Co., Ltd.
Don Mills, Ontario, Canada
M3B 1M2

02606
Fenwal Labs
Div. of Travenal Labs.
Morton Grove, Illinois

02660
Bunker Ramo Corp., Conn Div.
Formerly Amphenol-Borg
Electric Corp.
Broadview, Illinois

02799
Areo Capacitors, Inc.
Chatsworth, California

03508
General Electric Co.
Semiconductor Products
Syracuse, New York

03614
Replaced by 71400

03651
Replaced by 44655

03797
Eldema Div.
Genisco Technology Corp.
Compton, California

03877
Transistron Electronic Corp.
Wakefield, Massachusetts

03888
KDI Pyrofilm Corp.
Whippany, New Jersey

03911
Clairex Electronics Div.
Clairex Corp.
Mt. Vernon, New York

03980
Muirhead Inc.
Mountainside, New Jersey

04009
Arrow Hart Inc.
Hartford, Connecticut

04062
Replaced by 72136

04202
Replaced by 81312

04217
Essex International Inc.
Wire & Cable Div.
Anaheim, California

04221
Aemco, Div. of
Midtex Inc.
Mankato, Minnesota

04222
AVX Ceramics Div.
AVX Corp.
Myrtle Beach, Florida

04423
Telonic Industries
Laguna Beach, California

04645
Replaced by 75376

04713
Motorola Inc. Semiconductor
Products
Phoenix, Arizona

04946
Standard Wire & Cable
Los Angeles, California

05082
Replaced by 94988

05236
Jonathan Mfg. Co.
Fullerton, California

05245
Components Corp. now
Corcom, Inc.
Chicago, Illinois

05277
Westinghouse Electric Corp.
Semiconductor Div.
Youngwood, Pennsylvania

05278
Replaced by 43543

05279
Southwest Machine &
Plastic Co.
Glendora, California

05397
Union Carbide Corp.
Materials Systems Div.
New York, New York

05571
Use 56289
Sprague Electric Co.
Pacific Div.
Los Angeles, California

05574
Viking Industries
Chatsworth, California

05704
Replaced by 16258

05820
Wakefield Engineering Inc.
Wakefield, Massachusetts

06001
General Electric Co.
Electronic Capacitor &
Battery Products Dept.
Columbia, South Carolina

06136
Replaced by 63743

06383
Panduit Corp.
Tinley Park, Illinois

06473
Bunker Ramo Corp.
Amphenol SAMS Div.
Chatsworth, California

06555
Beede Electrical Instrument Co.
Penacook, New Hampshire

06739
Electron Corp.
Littleton, Colorado

06743
Clevite Corp.
Cleveland, Ohio

06751
Components, Inc. Semcor Div.
Phoenix, Arizona

06860
Gould Automotive Div.
City of Industry, California

06961
Vernitron Corp., Piezo
Electric Div.
Formerly Clevite Corp., Piezo
Electric Div.
Bedford, Ohio

06980
Eimac Div.
Varian Associates
San Carlos, California

07047
The Ross Milton Co.
South Hampton, Pennsylvania

07115
Replaced by 14674

07138
Westinghouse Electric Corp.,
Electronic Tube Div.
Horsehead, New York

07233
TRW Electronic Components
Cinch Graphic
City of Industry, California

07256
Silicon Transistor Corp.
Div. of BBF Group Inc.
Chelmsford, Massachusetts

07261
Aumet Corp.
Culver City, California

07263
Fairchild Semiconductor
Div. of Fairchild Camera
& Instrument Corp.
Mountain View, California

07344
Bircher Co., Inc.
Rochester, New York

07597
Burndy Corp.
Tape/Cable Div.
Rochester, New York

07792
Lerma Engineering Corp.
Northampton, Massachusetts

07910
Teledyne Semiconductor
Formerly Continental Device
Hawthorne, California

07933
Use 49956
Raytheon Co.
Semiconductor Div. HQ
Mountain View, California

08225
Industro Transistor Corp.
Long Island City, New York

Federal Supply Codes for Manufacturers (cont)

08261
Spectra Strip Corp.
Garden Grove, California

08530
Reliance Mica Corp.
Brooklyn, New York

08806
General Electric Co.
Miniature Lamp Products Dept.
Cleveland, Ohio

08863
Nylomatic Corp.
Norrisville, Pennsylvania

08988
Use 53085
Skottie Electronics Inc.
Archbald, Pennsylvania

09214
G.E. Co. Semi-Conductor
Products Dept.
Power Semi-Conductor
Products OPN Sec.
Auburn, New York

09353
C and K Components
Watertown, Massachusetts

09423
Scientific Components, Inc.
Santa Barbara, California

09922
Burndy Corp.
Norwalk, Connecticut

09969
Dale Electronics Inc.
Yankton, S. Dakota

10059
Barker Engineering Corp.
Formerly Amerace, Amerace
ESNA Corp.
Kenilworth, New Jersey

11236
CTS of Berne
Berne, Indiana

11237
CTS Keene Inc.
Paso Robles, California

11358
CBS Electronic Div.
Columbia Broadcasting System
Newburyport, Minnesota

11403
Best Products Co.
Chicago, Illinois

11503
Keystone Columbia Inc.
Warren, Michigan

11532
Teledyne Relays
Hawthorne, California

11711
General Instrument Corp
Rectifier Division
Hicksville, New York

11726
Qualidyne Corp.
Santa Clara, California

12014
Chicago Rivet & Machine Co.
Bellwood, Illinois

12040
National Semiconductor Corp.
Danburry, Connecticut

12060
Diodes, Inc.
Chatsworth, California

12136
Philadelphia Handle Co.
Camden, New Jersey

12300
Potter-Brumfield Div.
AMF Canada LTD.
Guelph, Ontario, Canada

12323
Presin Co., Inc.
Shelton, Connecticut

12327
Freeway Corp. formerly
Freeway Washer & Stamping Co.
Cleveland, Ohio

12443
The Budd Co. Polychem Products
Plastic Products Div.
Bridgeport, Pennsylvania

12615
U.S. Terminals Inc.
Cincinnati, Ohio

12617
Hamlin Inc.
Lake Mills, Wisconsin

12697
Clarostat Mfg. Co.
Dover, New Hampshire

12749
James Electronics
Chicago, Illinois

12856
Micrometals
Sierra Madre, California

12954
Dickson Electronics Corp.
Scottsdale, Arizona

12969
Unitrode Corp.
Watertown, Massachusetts

13103
Thermalloy Co., Inc.
Dallas, Texas

13327
Solitron Devices Inc.
Tappan, New York

13511
Amphenol Cadre Div
Bunker-Ramo Corp
Los Gatos, California

13606
Use 56289
Sprague Electric Co.
Transistor Div.
Concord, New Hampshire

13839
Replaced by 23732

14099
Semtech Corp.
Newbury Park, California

14140
Edison Electronic Div.
Mc Gray-Edison Co.
Manchester, New Hampshire

14193
Cal-R-Inc. formerly
California Resistor, Corp.
Santa Monica, California

14298
American Components, Inc.
an Insilco Co.
Conshohocken, Pennsylvania

14655
Cornell-Dublier Electronics
Division of Federal Pacific
Electric Co. Govt. Control Dept.
Newark, New Jersey

14752
Electro Cube Inc.
San Gabriel, California

14869
Replaced by 96853

14936
General Instrument Corp.
Semi Conductor Products Group
Hicksville, New York

15636
Elec-Trol Inc.
Saugus, California

15801
Fenwal Electronics Inc.
Div. of Kidde Walter and Co., Inc.
Framingham, Massachusetts

15818
Teledyne Semiconductors,
formerly Amelco Semiconductor
Mountain View, California

15849
Litton Systems Inc. Useco Div.
formerly Useco Inc.
Van Nuys, California

15898
International Business
Machines Corp.
Essex Junction, Vermont

15909
Replaced by 14140

16258
Space-Lok Inc.
Burbank, California

16299
Corning Glass
Electronic Components Div.
Raleigh, North Carolina

16332
Replaced by 28478

16473
Cambridge Scientific Ind.
Div. of Chemed Corporation
Cambridge, Maryland

16742
Paramount Plastics
Fabricators, Inc.
Downey, California

16758
Delco Electronics
Div. of General Motors Corp.
Kokomo, Indiana

17001
Replaced by 71468

17069
Circuit Structures Lab.
Burbank, California

17338
High Pressure Eng. Co., Inc.
Oklahoma City, Oklahoma

17545
Atlantic Semiconductors, Inc.
Asbury Park, New Jersey

17856
Siliconix, Inc.
Santa Clara, California

17870
Replaced by 14140

18178
Vactec Inc.
Maryland Heights, Missouri

18324
Signetics Corp.
Sunnyvale, California

18612
Vishay Resistor Products Div.
Vishay Intertechnology Inc.
Malvern, Pennsylvania

18736
Voltronics Corp.
Hanover, New Jersey

18927
GTE Sylvania Inc.
Precision Material Group
Parts Division
Titusville, Pennsylvania

19451
Perine Machinery & Supply Co.
Seattle, Washington

19701
Electro-Midland Corp.
Mepco-Electra Inc.
Mineral Wells, Texas

20584
Enochs Mfg. Inc.
Indianapolis, Indiana

Federal Supply Codes for Manufacturers (cont)

20891
Self-Organizing Systems, Inc.
Dallas, Texas

21604
Bucheye Stamping Co.
Columbus, Ohio

21845
Solitron Devices Inc.
Transistor Division
Riveria Beach, Florida

22767
ITT Semiconductors
Palo Alto, California

23050
Product Comp. Corp.
Mount Vernon, New York

23732
Tracor Inc.
Rockville, Maryland

23880
Stanford Applied Engrng.
Santa Clara, California

23936
Pamotor Div., Wm. J. Purdy Co.
Burlingame, California

24248
Replaced by 94222

24355
Analog Devices Inc.
Norwood, Massachusetts

24655
General Radio
Concord, Massachusetts

24759
Lenox-Fugle Electronics Inc.
South Plainfield, New Jersey

25088
Siemen Corp.
Isilen, New Jersey

25403
Amperex Electronic Corp.
Semiconductor &
Micro-Circuits Div.
Slatersville, Rhode Island

27014
National Semiconductor Corp.
Santa Clara, California

27264
Molex Products
Downers Grove, Illinois

28213
Minnesota Mining & Mfg. Co.
Consumer Products Div.
St. Paul, Minnesota

28425
Serv-/-Link formerly
Bohannan Industries
Fort Worth, Texas

28478
Deltrol Controls Div.
Deltrol Corporation
Milwaukee, Wisconsin

28480
Hewlett Packard Co.
Corporate HQ
Palo Alto, California

28520
Heyman Mfg. Co.
Kenilworth, New Jersey

29083
Monsanto, Co., Inc.
Santa Clara, California

29604
Stackpole Components Co.
Raleigh, North Carolina

30148
AB Enterprise Inc.
Ahoskie, North Carolina

30323
Illinois Tool Works, Inc.
Chicago, Illinois

31091
Optimax Inc.
Colmar, Pennsylvania

32539
Mura Corp.
Great Neck, New York

32767
Griffith Plastic Corp.
Burlingame, California

32879
Advanced Mechanical
Components
Northridge, California

32897
Erie Technological Products, Inc.
Frequency Control Div.
Carlisle, Pennsylvania

32997
Bourns Inc.
Trimpot Products Division
Riverside, California

33173
General Electric Co.
Products Dept.
Owensboro, Kentucky

34333
Silicon General
Westminister, California

34335
Advanced Micro Devices
Sunnyvale, California

34802
Electromotive Inc.
Kenilworth, New Jersey

37942
P.R. Mallory & Co., Inc.
Indianapolis, Indiana

42498
National Radio
Melrose, Massachusetts

43543
Nytronics Inc.
Transformer Co. Div.
Geneva, New York

44655
Ohmite Mfg. Co.
Skokie, Illinois

49671
RCA Corp.
New York, New York

49956
Raytheon Company
Lexington, Massachusetts

50088
Mostek Corp.
Carrollton, Texas

50579
Litronix Inc.
Cupertino, California

51605
Scientific Components Inc.
Linden, New Jersey

53021
Sangamo Electric Co.
Springfield, Illinois

54294
Cutler-Hammer Inc. formerly
Shallcross, A Cutter-Hammer Co.
Selma, North Carolina

55026
Simpson Electric Co.
Div. of Am. Gage and Mach. Co.
Elgin, Illinois

56289
Sprague Electric Co.
North Adams, Massachusetts

58474
Superior Electric Co.
Bristol, Connecticut

60399
Torin Corp. formerly
Torrington Mfg. Co.
Torrington, Connecticut

63743
Ward Leonard Electric Co., Inc.
Mount Vernon, New York

64834
West Mfg. Co.
San Francisco, California

65092
Weston Instruments Inc.
Newark, New Jersey

66150
Winslow Tele-Tronics Inc.
Eaton Town, New Jersey

70485
Atlantic India Rubber Works
Chicago, Illinois

70563
Amperite Company
Union City, New Jersey

70903
Belden Corp.
Geneva, Illinois

71002
Birnback Radio Co., Inc.
Freeport, New York

71400
Bussmann Mfg.
Div. of McGraw-Edison Co.
Saint Louis, Missouri

71450
CTS Corp.
Elkhart, Indiana

71468
ITT Cannon Electric Inc.
Santa Ana, California

71482
Clare, C.P. & Co.
Chicago, Illinois

71590
Centrelab Electronics
Div. of Globe Union Inc.
Milwaukee, Wisconsin

71707
Coto Coil Co., Inc.
Providence, Rhode Island

71744
Chicago Miniature Lamp Works
Chicago, Illinois

71785
TRW Electronics Components
Cinch Connector Operations Div.
Elk Grove Village
Chicago, Illinois

72005
Wilber B. Driver Co.
Newark, New Jersey

72092
Replaced by 06980

72136
Electro Motive Mfg. Co.
Williamantic, Connecticut

72259
Nytronics Inc.
Pelham Manor, New Jersey

72619
Dialight Div.
Amperex Electronic Corp.
Brooklyn, New York

72653
G.C. Electronics
Div. of Hydrometals, Inc.
Brooklyn, New York

72665
Replaced by 90303

72794
Dzus Fastener Co., Inc.
West Islip, New York

72928
Gulton Ind. Inc.
Gudeman Div.
Chicago, Illinois

Federal Supply Codes for Manufacturers (cont)

72982
Erie Tech. Products Inc.
Erie, Pennsylvania

73138
Bechman Instrument Inc.
Helipot Division
Fullerton, California

73293
Hughes Aircraft Co.
Electron Dynamics Div.
Torrance, California

73445
Amperex Electronic Corp.
Hicksville, New York

73559
Carling Electric Inc.
West Hartford, Connecticut

73586
Circle F Industries
Trenton, New Jersey

73734
Federal Screw Products, Inc.
Chicago, Illinois

73743
Fischer Special Mfg. Co.
Cincinnati, Ohio

73899
JFD Electronics Co.
Components Corp.
Brooklyn, New York

73949
Guardian Electric Mfg. Co.
Chicago, Illinois

74199
Quan Nichols Co.
Chicago, Illinois

74217
Radio Switch Corp.
Marlboro, New Jersey

74276
Signalite Div.
General Instrument Corp.
Neptune, New Jersey

74306
Piezo Crystal Co.
Carlisle, Pennsylvania

74542
Hoyt Elect. Instr. Works
Penacook, New Hampshire

74970
Johnson E.F., Co.
Waseca, Minnesota

75042
TRW Electronics Components
IRC Fixed Resistors
Philadelphia, Pennsylvania

75376
Kurz-Kasch Inc.
Dayton, Ohio

75378
CTS Knights Inc.
Sandwich, Illinois

75382
Kulka Electric Corp.
Mount Vernon, New York

75915
Littlefuse Inc.
Des Plaines, Illinois

76854
Oak Industries Inc.
Switch Div.
Crystal Lake, Illinois

77342
AMF Inc.
Potter & Brumfield Div.
Princeton, Indiana

77638
General Instrument Corp.
Rectifier Division
Brooklyn, New York

77969
Rubbercraft Corp. of CA. LTD.
Torrance, California

78189
Shakeproof
Div. of Illinois Tool Works Inc.
Elgin, Illinois

78277
Sigma Instruments, Inc.
South Braintree, Massachusetts

78488
Stackpole Carbon Co.
Saint Marys, Pennsylvania

78553
Eaton Corp. Engineered
Fastener Div.
Tinnerman Plant
Cleveland, Ohio

79136
Waldes Kohinoor Inc.
Long Island City, New York

79497
Western Rubber Company
Goshen, Indiana

79963
Zierick Mfg. Corp.
Mt. Kisko, New York

80031
Electro-Midland Corp.
Mepco Div.
A North American Phillips Co.
Norristown, New Jersey

80145
LFE Corp., Process Control Div.
formerly API Instrument Co.
Chesterland, Ohio

80183
Use 56289
Sprague Products
North Adams, Massachusetts

80294
Bourns Inc., Instrument Div.
Riverside, California

80583
Hammarlund Mfg. Co., Inc.
Red Bank, New Jersey

80640
Arnold Stevens, Inc.
South Boston, Massachusetts

81073
Grayhill, Inc.
La Grange, Illinois

81312
Winchester Electronics
Div. of Litton Industries Inc.
Oakville, Connecticut

81483
Therm-O-Disc Inc.
Mansfield, Ohio

81483
International Rectifier Corp.
Los Angeles, California

81590
Korry Mfg. Co.
Seattle, Washington

81741
Chicago Lock Co.
Chicago, Illinois

82305
Palmer Electronics Corp.
South Gate, California

82389
Switchcraft Inc.
Chicago, Illinois

82415
North American Phillips
Controls Corp.
Frederick, Maryland

82872
Roanwell Corp.
New York, New York

82877
Rotron Inc.
Woodstock, New York

82879
ITT Royal Electric Div.
Pawtucket, Rhode Island

83003
Varo Inc.
Garland, Texas

83058
The Carr Co., United Can Div.
of TRW
Cambridge, Massachusetts

83298
Bendix Corp.
Electric Power Div.
Eatontown, New Jersey

83330
Herman H. Smith, Inc.
Brooklyn, New York

83478
Rubbercraft Corp
of America, Inc.
West Haven, Connecticut

83594
Burroughs Corp.
Electronic Components Div.
Plainfield, New Jersey

83740
Union Carbide Corp.
Battery Products Div.
formerly Consumer Products Div.
New York, New York

84171
Arco Electronics
Great Neck, New York

84411
TRW Electronic Components
TRW Capacitors
Ogallala, Nebraska

84613
Fuse Indicator Corp.
Rockville, Maryland

84682
Essex International Inc.
Industrial Wire Div.
Peabody, Massachusetts

86577
Precision Metal Products
of Malden Inc.
Stoneham, Massachusetts

86684
Radio Corp. of America
Electronic Components Div.
Harrison, New Jersey

86928
Seastrom Mfg. Co., Inc.
Glendale, California

87034
Illuminated Products Inc.
Subsidiary of Oak Industries Inc.
Anahiem, California

88219
Gould Inc.
Industrial Div.
Trenton, New Jersey

88245
Litton Systems Inc.
Useco Div.
Van Nuys, California

88419
Cornell-Dubilier Electronic Div.
Federal Pacific Co.
Fuquay-Varian, North Carolina

88486
Plastic Wire & Cable
Jewitt City, Connecticut

88690
Replaced by 04217

89536
John Fluke Mfg. Co., Inc.
Seattle, Washington

89730
G.E. Co., Newark Lamp Works
Newark, New Jersey

Federal Supply Codes for Manufacturers (cont)

90201
Mallory Capacitor Co.
Div. of P.R. Mallory Co., Inc.
Indianapolis, Indiana

90211
Use 56365
Square D Co.
Chicago, Illinois

90215
Best Stamp & Mfg. Co.
Kansas City, Missouri

90303
Mallory Battery Co.
Div. of Mallory Co., Inc.
Tarrytown, New York

91094
Essex International Inc.
Suglex/IWP Div.
Newmarket, New Hampshire

91293
Johanson Mfg. Co.
Boonton, New Jersey

91407
Replaced by 58474

91502
Associated Machine
Santa Clara, California

91506
Augat Inc.
Attleboro, Massachusetts

91637
Dale Electronics Inc.
Columbus, Nebraska

91662
Elco Corp.
Willow Grove, Pennsylvania

91737
Use 71468
Gremar Mfg. Co., Inc.
ITT Cannon/Gremar
Santa Ana, California

91802
Industrial Devices, Inc.
Edgewater, New Jersey

91833
Keystone Electronics Corp.
New York, New York

91836
King's Electronics Co., Inc.
Tuckahoe, New York

91929
Honeywell Inc.
Micro Switch Div.
Freeport, Illinois

91934
Miller Electric Co., Inc.
Div. of Aunet
Woonsocket, Rhode Island

92194
Alpha Wire Corp.
Elizabeth, New Jersey

93332
Sylvania Electric Products
Semiconductor Products Div.
Woburn, Massachusetts

94145
Replaced by 49956

94154
Use 94988
Wagner Electric Corp.
Tung-Sol Div.
Newark, New Jersey

94222
Southco Inc. formerly
South Chester Corp.
Lester, Pennsylvania

95146
Alco Electronic Products Inc.
Lawrence, Massachusetts

95263
Leecraft Mfg. Co.
Long Island City, New York

95264
Replaced by 98278

95275
Vitramon Inc.
Bridgeport, Connecticut

95303
RCA Corp.
Receiving Tube Div.
Cincinnati, Ohio

95348
Gordo's Corp.
Bloomfield, New Jersey

95354
Methode Mfg. Corp.
Rolling Meadows, Illinois

95712
Bendix Corp.
Electrical Components Div.
Microwave Devices Plant
Franklin, Indiana

95987
Weckesser Co. Inc.
Chicago, Illinois

96733
San Fernando Electric Mfg. Co.
San Fernando, California

96853
Gulton Industries Inc.
Measurement and Controls Div.
formerly Rustrak Instruments Co.
Manchester, New Hampshire

96881
Thomson Industries, Inc.
Manhasset, New York

97540
Master Mobile Mounts, Div. of
Whitehall Electronics Corp.
Ft. Meyers, Florida

97913
Industrial Electronic
Hardware Corp.
New York, New York

97945
Penwalt Corp.
SS White Industrial Products Div.
Piscataway, New Jersey

97966
Replaced by 11358

98094
Replaced by 49956

98159
Rubber-Teck, Inc.
Gardena, California

98278
Malco A Microdot Co., Inc.
Connector & Cable Div.
Pasadena, California

98291
Sealectro Corp.
Mamaroneck, New York

98388
Royal Industries
Products Div.
San Diego, California

98743
Replaced by 12749

98925
Replaced by 14433

99120
Plastic Capacitors, Inc.
Chicago, Illinois

99217
Bell Industries Elect.
Comp. Div.
formerly Southern Elect. Div.
Burbank, California

99392
STM
Oakland, California

99515
ITT Jennings Monrovia Plant
Div. of ITT Jennings formerly
Marshall Industries Capacitor Div.
Monrovia, California

99779
Use 29587
Bunker-Ramo Corp.
Barnes Div.
Landsdowne, Pennsylvania

99800
American Precision Industries Inc.
Delevan Division
East Aurora, New York

99942
Centrelab Semiconductor
Centrelab Electronics Div. of
Globe-Union Inc.
El Monte, California

Toyo Electronics
(R-Ohm Corp.)
Irvine, California

National Connector
Minneapolis, Minnesota

Fluke Technical Service Centers - U.S. and Canada

UNITED STATES

CA, Burbank
Fluke Technical Center
2020 N. Lincoln Street
Burbank, CA 91504
(213) 849-4641

CA, Santa Clara
Fluke Technical Center
2300 Walsh Avenue
Santa Clara, CA 95050
(408) 985-1200

CO, Denver
Fluke Technical Center
1980 S. Quebec Street, Unit 4
Denver, CO 80231
(303) 750-1228

FL, Orlando
Fluke Technical Center
940 N. Fern Creek Avenue
Orlando, FL 32803
(305) 896-2296

IL, Rolling Meadows
Fluke Technical Center
1400 Hicks Road
Rolling Meadows, IL 60008
(312) 398-5800

MA, Waltham
Fluke Technical Center
244 Second Avenue
Waltham, MA 02154
(617) 890-1604

MD, Rockville
Fluke Technical Center
5640 Fishers Lane
Rockville, MD 20852
(301) 770-1576

MN, Apple Valley
Fluke Technical Center
7373 West 147th Street, Suite 196
Apple Valley, MN 55124
(612) 432-9400

NC, Greensboro
Fluke Technical Center
1310 Beaman Place
Greensboro, NC 27408
(919) 273-1918

NJ, Paramus
Fluke Technical Center
West 75th Century Road
Paramus, NJ 07652
(210) 262-9550

TX, Dallas
Fluke Technical Center
14400 Midway Road
Dallas, TX 75240
(214) 233-9945

WA, Mountlake Terrace
John Fluke Mfg. Co., Inc.
21707 66th Avenue W., Suite 1
Mountlake Terrace, WA 98043
(206) 774-2206

CANADA

ALB, Calgary
Allan Crawford Assoc., Ltd.
1935 30th Avenue N.E. #14
Calgary, ALB T2E 6Z5
(403) 230-1341

ONT, Mississauga
Allan Crawford Assoc., Ltd.
6503 Northam Drive
Mississauga, ONT L4V 1J5
(416) 678-1500

QUE, St. Laurent
Allan Crawford Assoc., Ltd.
7018 Cote de Liesse
St. Laurent, QUE H4T 1E7
(514) 731-8564

Technical Service Centers - International

ARGENTINA
Fluke Latin American
Service Headquarters
Virrey del Pino 4071
Buenos Aries, Argentina
Tel: 523185

AUSTRALIA
Elmeasco Instrument Pty Ltd.
P.O. Box 30
Concord, N.S.W.
Australia 2137
Tel: (02) 736-2888

AUSTRIA
Walter Rekirsch
Elektronische Gerate GmbH & Co.
Vertriebs-KG.
Liechtensteinstrasse 97/6
A-1090 Vienna, Austria
Tel: 09-43-222-347646

BELGIUM
Fluke (Belgium)S.A./N.V.
6, Rue de Geneve
1140 Brussels, Belgium
Tel: 09-32-2-2164090

BRAZIL
Arotec S.A
Industria e Comercio
Av. Pacaembu 811
01234 Sao Paulo S.P , Brazil
Tel. 826-2266

CHILE
Intronica Chile Ltda.
Casilla 16228
Santiago 9, Chile
Tel: 44940

COLOMBIA
Coasin Ltda.
Carrera 13, # 37-37, Of. 407
Ap. Aero 29583
Bogota DE, Colombia
Tel: 285-0230

DENMARK
Tage Olsen A/S
Ballerup Byvej 222
DK-2750 Ballerup
Denmark
Tel: (01) 2-65 81 11

ECUADOR
Proteco Coasin Cia., Ltda.
Edificia "Jerico"
Ave. 12 de Octubre #2285y
Ave. Orellana (Planta Baja)
Quito, Ecuador
Tel: 526759

EGYPT
Lotus Engineering Organisation
P.O. Box 1252
Cairo, Egypt
Tel: 71617

FINLAND
Oy Findip AB
Teollisuustie 7
02700 Kauniainen
Finland
Tel: 09-358-0-502255

FRANCE
M.B. Electronique S.A
Rue Fourny
ZAC de BUC
B.P. #31
78530 BUC, France
Tel: 09-33-1-9568131

GREECE
Hellenic Scientific
Representations Ltd.
11, Vrassida Street
Athens 615, Greece
Tel: 09-30-1-711140

HONG KONG
Gilman Industrial Bldg., 9F
107-109 Wai Yip Street
Kwun Tong
Kowloon, Hong Kong
Tel: 3-427144

INDIA
Hinditron Services Pvt. Ltd.
"Hinditron House"
412 Raj Mahal Vilas Extension
Bangalore 560 006, India
Tel: 33139

Hinditron Services Pvt. Ltd.
69 A.L. Jagmohandas Marg
Bombay 400 006, India
Tel: 365344

INDONESIA
P.T. Dwi Tunggal Jaya Sakti
Sangga Buana Bldg., 1st Floor
J1. Senen Raya 44, P.O. Box 4435
Jakarta, Indonesia
Tel: 376608

IRAN
Irantronics Company Ltd
20 Salm Road
Roosevelt Avenue
Tehran, Iran
Tel: 828294/831564

ISRAEL
R.D.T. Electronics Engineering Ltd.
P.O. Box 75
46, Sokolov Street
Ramat Hasharon 47235
Israel
Tel: 09-972-3-483211

ITALY
Sistrel S.p.A.
Via Timavo 66
20099 Sesto S. Giovanni (Milan)
Italy
Tel: 09-39-2-2476693

Sistrel S.P.A.
Via Guiseppe Armellini #37
00143 Rome, Italy
Tel: 09-39-6-5915551

JAPAN
John Fluke Mfg. Co., Inc.
Japan Branch
1 Higashikata-machi
Midori-ku
Yokohama 226, Japan
Tel: (045) 473-5425

KOREA
Electro-Science Korea Co.
C.P.O. Box 8446
Room 1201 Bowon Bldg
490 Chongro - 5 Ka
Chongro-ku
Seoul, Korea
Tel. 261-7702

MALASIA
O'Connor's (Malaysia) SDN. BHD.
P.O. Box 91
Petaling Jaya, Selangor
West Malaysia
Tel: 51563

Technical Service Centers - International (cont)

MEXICO
C.J. Christensen S.A. De C.V.
Melchor Ocampo 150-8
Mexico 4 D.F., Mexico
Tel: (905) 535-2258

NETHERLANDS
Fluke (Nederland) B.V.
P.O. Box 5053
5004 EB Tilburg
The Netherlands
Tel: (013) 673973

NEW ZEALAND
W & K McLean Ltd.
P.O. Box 18065
Glen Innes
Auchland, New Zealand
Tel: 587-037

NORWAY
Morgenstierne & Co. A/A
Konghellegate 3
P.O. Box 6688, Rodelokka
Oslo 5, Norway
Tel: 09-47-2-356110

PAKISTAN
Pak International Operations
505 Mohammadi House
1.1. Chundrigar Road
Katrachi-2 Pakistan
Tel: 221127/239052

PERU
Importaciones Y Representaciones Electronicas S.A.
Avda. Franklin D. Roosevelt 105
Lima 1, Peru
Tel: 288650

PORTUGAL
Equipamentos de Laboratorio, Ltda
P.O. Box 1100
Lisbon 1002, Portugal
Tel: 09-351-1-578936/547512

REPUBLIC OF SINGAPORE
Rank O'Connor's (PTE) Ltd.
98 Pasir Panjang Road
Singapore 0511
Republic of Singapore
Tel: 637944

REPUBLIC OF SOUTH AFRICA
Fluke S.A. (Pty) Ltd.
P.O. Box 39797
Bramley 2018
Republic of South Africa
Tel: (011) 786-3170

SPAIN
Hispano Electronica S.A.
Apartado de Correos 48
Alcorcon (Madrid)
Spain
Tel: 09-3416194108

SWEDEN
Teleinstrument AB
P.O. Box 490
S-162 Vallingby 4
Sweden
Tel: 09-46-8-380370

SWITZERLAND
Traco Electronic AG
Jenatschstrasse 1
8002 Zurich
Switzerland
Tel: 09-41-1-2010711

TAIWAN
CCT Associates, Inc.
P.O. Box 24209
Taipei, Taiwan
Republic of China
Tel: (02) 391-6894/393-5760

THAILAND
Dynamic Supply Engineering R.O.P.
56 Ekamai, Sukhumvit 63
Bangkok-11, Thailand
Tel: 914434/928532

TURKEY
Erkman Electronik Aletler
Ticaret Anonim Sirketi
Necatiby Cad 92/3
Karakoy, Istanbul, Turkey
Tel: 09-44-1-5461

UNITED KINGDOM
Fluke International Corp. - UK
Colonial Way
Watford, Herts. WD24TT
United Kingdom
Tel: 09-4492340511

URUGUAY
Coasin Uruguaya S.R.L.
Edifico Ciudadela
Sarandi 690 D. Esc. 109
Casilla de Correos 1400
Correo Central
Montevideo, Uruguay
Tel: 917978

VENEZUELA
Coasin C.A.
Apdo. Postal 50939
Sabana Grande # 1
Caracas 105, Venezuela
Tel: 782-9109

WEST GERMANY
Fluke (Deutschland) GmbH
Krausstrasse 32
8045 Ismaning
West Germany
Tel: (089) 96251

Customers in the following countries: Bulgaria, Czechoslovakia, Hungary, Poland, Romania, U.S.S.R., and Yugoslavia, contact: Amtest Associates Ltd., P.O. Box 55, Addlestone, Surrey, KT 15 1DU, England, Tel. (0932) 52121

Sales Representatives - U.S. and Canada

UNITED STATES

AK, Anchorage
Harry Lang & Associates
1371 Hillcrest Drive #303
Anchorage, AK 99503
(907) 279-5741

AL, Huntsville
John Fluke Mfg. Co., Inc.
3322 Memorial Parkway
Huntsville, AL 35807
(205) 881-6220
(404) 321-0980 (Atlanta)

AZ, Tempe
John Fluke Mfg. Co., Inc.
2125 S. 48th Street, Suite 104
Tempe, AZ 85285
(602) 967-8724

CA, Los Angeles
John Fluke Mfg. Co., Inc.
20902 S. Bonita Street
Carson, CA 90746
(213) 538-3900

CA, Santa Clara
John Fluke Mfg. Co., Inc.
2300 Walsh Avenue
Santa Clara, CA 95050
(408) 244-1505

CA, Tustin
John Fluke Mfg. Co., Inc.
15445 Red Hill Avenue, Suite F
Tustin, CA 92680
(714) 838-8863
(714) 226-1254 (San Diego)

CO, Denver
John Fluke Mfg. Co., Inc.
1980 S. Quebec, #4
Denver, CO 80231
(303) 750-1222

CT, Hartford
John Fluke Mfg. Co., Inc.
124 Hebron Avenue
Glastonbury, CT 06033
(203) 633-0777

FL, Orlando
John Fluke Mfg. Co., Inc.
940 N. Fern Creek Avenue
Orlando, FL 32803
(305) 896-4881

HI, Honolulu
EMC Corporation
2979 Ualena Street
Honolulu, HI 96819
(808) 847-1138

IA, Iowa City
John Fluke Mfg. Co., Inc.
1027 Hollywood Blvd., Suite 309
Iowa City, IA 52240
(319) 354-2811

IL, Chicago
John Fluke Mfg. Co., Inc.
4300 Lincoln, Unit K
Rolling Meadows, IL 60008
(312) 398-0850

IN, Indianapolis
John Fluke Mfg. Co., Inc.
5610 Crawfordsville Road,
Suite 802
Indianapolis, IN 46224
(317) 244-2456

LA, Metairie
John Fluke Mfg. Co., Inc.
2700 Kingman Street, Suite 103
Metairie, LA 70002
(504) 455-0814

MA, Waltham
John Fluke Mfg. Co., Inc.
244 Second Avenue
Waltham, MA 02154
(617) 890-1600

MD, Baltimore
John Fluke Mfg. Co., Inc.
5640 Fishers Lane
Rockville, MD 20852
(301) 770-1570
(301) 792-7060(Baltimore)

MI, Detroit
John Fluke Mfg. Co., Inc.
13955 Farmington Road
Livonia, MI 48154
(313) 522-9140

MN, Minneapolis
John Fluke Mfg. Co., Inc.
7373 W. 147th Street, Suite 196
Apple Valley, MN 55124
(612) 432-9400

MO, Kansas City
John Fluke Mfg. Co., Inc.
4406 Chouteau Traffic Way
Kansas City, MO 64117
(816) 454-5836

MO, St. Louis
John Fluke Mfg. Co., Inc.
300 Brookes Drive, Suite 100
Hazelwood, MO 63042
(314) 731-3388

Sales Representatives - U.S. and Canada (cont)

NC, Greensboro
John Fluke Mfg. Co., Inc.
1310 Beaman Place
Greensboro, NC 27408
(919) 273-1918

NJ, Paramus
John Fluke Mfg. Co., Inc.
West 75 Century Road
Paramus, NJ 07625
(201) 262-9550

NM, Albuquerque
John Fluke Mfg. Co., Inc.
1108 Alvarado Drive N.E.
Albuquerque, NM 87110
(505) 265-8431

NY, Rochester
John Fluke Mfg. Co., Inc.
4515 Culver Road
Rochester, NY 14622
(716) 266-1400

OH, Cleveland
John Fluke Mfg. Co., Inc.
7830 Freeway Circle
Middleburg Heights, OH 44130
(216) 234-4540

OH, Dayton
John Fluke Mfg. Co., Inc.
4756 Fishburg Road
Dayton, OH 45424
(513) 233-2238

OR, Portland
John Fluke Mfg. Co., Inc.
18360 S.W. Springfield Lane
Aloha, OR 97006
(502) 642-1342

PA, Philadelphia
John Fluke Mfg. Co., Inc.
1010 W. 8th Avenue, Suite H
King of Prussia, PA 19406
(215) 265-4040
(412) 261-5171 (Pittsburgh)

TX, Austin
John Fluke Mfg. Co., Inc.
111 W. Anderson Lane, Suite 213
Austin, TX 78752
(512) 478-9901

TX, Dallas
John Fluke Mfg. Co., Inc.
14400 Midway Road
Dallas, TX 75243
(214) 233-9990

TX, Houston
John Fluke Mfg. Co., Inc.
1014 Wirt Road, Suite 270
Houston, TX 77055
(713) 683-7913
(512) 222-2726 (San Antonio)

UT, Salt Lake City
John Fluke Mfg. Co., Inc.
782 E. 8325 South
Sandy, UT 84070
(801) 566-4864

WA, Kennewick
John Fluke Mfg. Co., Inc.
3613 W. 16th Place #C
Kennewick, WA 99336

WA, Seattle
John Fluke Mfg. Co., Inc.
975 Industry Drive
Seattle, WA 98188
(206) 575-3765

CANADA

ALB, Calgary
Allan Crawford Assoc., Ltd.
1935 30th Avenue N.E., #14
Calgary, ALB T2E 6Z5
Tel: (403) 230-1341

BC, North Vancouver
Allan Crawford Assoc., Ltd
3795 William Street
Burnaby, BC Y5C 3H3
Tel: (604) 294-1326

NS, Halifax
Allan Crawford Assoc., Ltd
Townsend Place, Suite 201
800 Windmill Road
Burnside Industrial Park
Dartmounth, NS B3B 1L1
Tel: (902) 469-7865

ONT, Ottawa
Allan Crawford Assoc., Ltd
1299 Richmond Road
Ottawa, ONT K2B 7Y4
Tel: (613) 829-9651

ONT, Toronto
Allan Crawford Assoc., Ltd.
6503 Northam Drive
Mississauga, ONT L4V 1J5
Tel: (416) 678-1500

QUE, Montreal
Allan Crawford Assoc., Ltd.
7018 Cote de Liesse
St. Laurent, QUE H4T 1E7
Tel: (514) 731-8564

For more information on Fluke products or Sales Offices you may dial (800) 426-0361 toll free in most of U.S. From Alaska, Hawaii, Washington, or Canada phone (206) 774-2481. From other countries phone (206) 774-2398.

For Canadian areas not listed, contact the office nearest you or Allan Crawford Assoc. Ltd., Mississauga (Toronto), Ontario.

Sales Representatives - International

Supplied and Supported by Fluke International Corporation, P.O. Box 43210, Mountlake Terrace, WA 98043, Tel: (206) 774-2398, TLX: 152662 JOHNFLUKE MTLT

ARGENTINA
Coasin S.A.
Virrey del Pino 4071
Buenos Aires, Argentina
Tel: 525130
TLX: 390122284 COASN AR

AUSTRALIA
Elmeasco Instruments Pty Ltd.
P.O. Box 30
Concord, N.S.W.
Australia 2137
Tel: (2) 736-2888
TLX: 79025887

Elmeasco Instruments Pty Ltd.
P.O. Box 107
Mt. Waverly, VIC. 3149
Australia
Tel: 233-4044
TLX: 26206 (ELMVIC)

Elmeasco Instruments Pty Ltd.
Professional Suite's Bldg.
B.P.O. Box 2360
Brisbane, 4001
Australia
Tel: (07) 229-3161

Elmeasco Instruments Pty Ltd.
P.O. Box 1240
G.P.O. Adelaide 5001
South Australia

BOLIVIA
Coasin Bolivia S.R.L.
Casilla 7295
La Paz, Bolivia
Tel: 40962
TLX: Public Booth # 5377, indicate "Para Coasin B.S.R.L., Av 6 de Agosto 2300 3er Piso, La Paz tl 40962"

BRAZIL
Arotec S.A.
Industria e Comercio
Av. Pacaembu 811
01234 Sao Paulo S.P., Brazil
Tel: (011) 67-2393
TLX: 1122207 AROT BR

Arotec S.A.
Rua Araguari, 1705 s 402
30.000 Belo Horizonte M.G., Brazil

Arotec S.A.
Av. Rio Branco 277
Grupo 1309
02000 Rio de Janeiro, R.J., Brazil
Tel: (021) 242-9776

BRUNEI
O'Connor's Limited
(Brunei Branch) Chin Kian Bldg.
P.O. Box 2126
Bandar Seri Begawan, Brunei
Tel: 26680

CHILE
Intronica Chile Ltda
Manuel Montt 024 - Of. D
Casilla 16228
Santiago 9, Chile
Tel: 44940
TLX: 40301 Cab Publ.,
Attn: Intronica Chile

CHINA, PEOPLE'S REPUBLIC OF
China National Instrument Import & Export Corp.
P.O. Box 49, Erh-Li-Kou
Hsi Chi'ao
Beijing
People's Republic of China
TLX: CO 2242 CNIEC

COLOMBIA
Coasin Ltda.
Carrera 13, # 37-37, Of. 407
Ap. Aero 29583
Bogota DE, Colombia
Tel: 285-0230/0250
TLX: 45787 COASN CO

ECUADOR
Proteco Coasin Cia., Ltda.
Edificio "Jerico"
Ave. 12 de Octubre #2285 y
Ave. Oreliana (Planta Baja)
Quito, Ecuador
Tel: 529684
TLX: 2865 Protec Ed

Proteco Coasin Cia., Ltda
Casilla 9733
9 de Octubre 424
Edificio Gran Pasaje, Of. 1012
Guayaquil, Ecuador

Sales Representatives - International (cont)

HONG KONG
Gilman & Co., Ltd.
P.O. Box 56
Gilman Ind. Bldg. 9 F
107-109 Wai Yip Street
Kwun Tong, Kowloon
Hong Kong
Tel: 5-7909633
TLX: 73358 GILMN HX

ICELAND
Kristjan O. Skagfjord Ltd.
P.O. Box 906
Reykjavik, Iceland
Tel: 24120
TLX: 2133 KOS-1S

INDIA
Hinditron Services Pvt. Ltd.
69 A.L., Jagmohandas Marg
Bombay 400 006
India
Tel: 811316
TLX: 953-112326 Hspl In

Hinditron Services Pvt. Ltd.
"Hinditron House"
412 Raj Mahl Vilas Extension
Bangalore 560 006
India
Tel: 33139
TLX: 043741

INDONESIA
P.T. Dwi Tunggal Jaya Sakti
Sangga Buana Bldg., 1st Floor
Jl. Senen Raya 44, P.O. Box 4435
Jakarta, Indonesia
Tel: 367390-9
TLX: 46624 SABUANA 1A

P.T. Dwi Tunggal Jaya Sakti
Jalan Sasakgantung 45
Bandung, Indonesia
Tel: 367390-9
TLX: 46624 SABUANA 1A

JAPAN
John Fluke Mfg. Co., Inc.
Japan Branch
(PC Board Testers only)
1 Higashikata-machi
Midori-ku
Yokohama 226, Japan
Tel: (045) 473-5425
TLX: 3823-666 FLUKJPJ

Tokyo Electron Ltd.
38 Fl. Shinjuko Nomura Bldg.
1-26-2, Nishi-Shinjuku
Shinjuku-ku
Tokoyo 160, Japan
Tel: 03-343-4411
TLX: 2322220 LABTEL J

KENYA
ADCom Limited Inc.
P.O. Box 30070
Nairobi, Kenya
East Africa
Tel: 331955
TLX: 22639 ADCOM

KOREA
Electro-Science Korea Co.
C.P.O. Box 8446
Room. 1201 Bowon Bldg.
490 Chongro - 5 Ka
Chongro-ku
Seoul, Korea
Tel: 261-7702
TLX: 78723270+

MALAYSIA
O'Connor's (PTE) Limited
P.O. Box 91
Petaling Jaya, Selangor
West Malaysia
Tel: 51563
TLX: OCONOR MA37649

O'Connor's (PTE) Limited
Lot # 5 Taman Mesra
Mile 3, Jalan Penampang
Kota Kinabulu
East Malaysia
Tel: 55322
TLX: MA80286FL

MEXICO
C.J. Christensen S.A. De C.V.
Instrumentos Electronicos
de Medicion
Melchor Ocampo 150-8
Mexico 4 D.F., Mexico
Tel: (905) 535-2258

NEW ZEALAND
W & K McLean Ltd.
P.O. Box 18065
Glen Innes
Auckland, New Zealand
Tel: 587-037
TLX: N.Z. 7912763

W & K McLean Ltd.
P.O. Box 2421
Christchurch, New Zealand

W & K McLean Ltd.
P.O. Box 496
Wellington, New Zealand
Tel: 851-450

NIGERIA
Mofat Engineering Co., Ltd.
P.O. Box 6369
Lagos, Nigeria
TLX: 21353, Attn: MOFAT

PAKISTAN
Pak International Operations
505 Mohammadi House
- McLeod Road
P.O. Box 5323
Karachi, Pakistan
Tel: 221127
TLX: 24494 PIO PK

PERU
Importaciones Y Representaciones
Electronicas S.A.
Avda. Franklin D. Roosevelt 105
Lima 1, Peru
Tel: 288650
TLX: 37425663

PHILIPPINES
Spark Radio and Electric Supply
P.O. Box 1333
Manila, Philippines
Tel: 473183
TLX: 27901 RLA PH

SINGAPORE
Rank O'Connor's (PTE) Limited
98 Pasir Panjang Road
Singapore 0511
Republic of Singapore
Tel: 637944
TLX: OCONSIN RS21023

SOUTH AFRICA
Fluke S.A. (Pty) Ltd.
P.O. Box 39797
Bramley 2018
Republic of South Africa
Tel: (011) 786-3170
TLX: 82899

TAIWAN
CCT Associates, Inc.
P.O. Box 24209
Taipei, Taiwan
Republic of China
Tel: (02) 391-6894
TLX: 24263 Longgulf

THAILAND
Dynamic Supply Engineering R.O.P.
56 Ekamai, Sukhumvit 63
Bangkok-11, Thailand
Tel: 914434
TLX: TH 82938 MONTIEN TH

URUGUAY
Coasin Uruguaya S.R.L.
Edif. Ciudadela
Sarandi 690 D. Esc. 109
Casilla de Correos 1400
Correo Central
Montevideo, Uruguay
Tel: 917978
TLX: Public Booth UY901 Coasin

VENEZUELA
Coasin C.A.
Apdo. Postal 50939
Sabana Grande # 1
Caracas 105, Venezuela
Tel: 782-9109-8741
TLX: 39531228

Latin America also Supported by -
Sr. Leonardo Cusnir, c/o COASIN A.A., Virrey del Pino 4071, Buenos Aires, Argentina, Tel: 523185, TLX: 390122284 COASN AR
Sr. Benewaldo Padovani, c/o AROTEC S.A., Industria e Comericio, Av. Pacaembu 811, 01234 Sao Paulo S.P., Brazil, Tel: (011) 67-2393, TLX: 1122207 AROT BR

Supplied and Supported by Fluke (Holland) B.V., P.O. Box 5053, Zevenheuvelenweg 53, 5004 EB Tilburg, Netherlands, Tel: (013) 673973, TLX: 522337

AUSTRIA
Walter Rekirsch
Electronische Gerate GmbH & Co.
Vertriebs-KG
Liechtensteinstrasse 97/6
A-1090 Vienna, Austria
Tel: (222) 347646
TLX: 134759

BELGIUM
Fluke (Belguim) S.A./N.V.
6 Rue de Geneve
1140 Brussels, Belgium
Tel: (2) 2164090
TLX: 26312

BULGARIA
Amtest Associates Ltd.
P.O. Box 55
Addlestone, Surrey KT15 1DU
United Kingdom
Tel: Weybridge (923) 52121
TLX: 928855

CHAD
Fluke (Holland) B.V.
P.O. Box 5053
Zevenheuvelenweg 53
5004 EB Tilburg, Netherlands
Tel: (013) 673973
TLX: 52237

CYPRUS
Chris Radiovision Ltd.
P.O. Box 1989
Nicosia, Cyprus
Tel: 66121
TLX: 2395

CZECHOSLOVAKIA
Amtest Associates Ltd.
P.O. Box 55
Addlestone, Surrey KT15 1DU
United Kingdom
Tel: Weybridge (932) 52121
TLX: 928855

DENMARK
Tage Olsen A/S
Ballerup Byvej 222
DK - 2750 Ballerup, Denmark
Tel: (2) 658111
TLX: 35293

EGYPT AND SUDAN
Lotus Engineering Organisation
P.O. Box 1252
22, Kasi El Nil
Cairo, Egypt
Tel: 971617
TLX: 92504

Sales Representatives - International (cont)

EIRE (REPUBLIC OF IRELAND)
Euro Electronics
32 Brews Hill
Naven, County Meath
Tel: 46-23577
TLX: 31821

ETHIOPIA
Fluke (Holland) B.V.
P.O. Box 5053
Zevenheuvelenweg 53
5004 EB Tilburg, Netherlands
Tel: (013) 673973
TLX: 52237

FINLAND
Oy Findip AB
Teollisuustie 7
02700 Kauniainen, Finland
Tel: (0) 502255
TLX: 123129

FRANCE
M.B. Elctronique S.A.
Rue Fourny
ZAC de BUC
B.P. # 31
78530 BUC, France
Tel: (1) 9568131
TLX: 695414

GERMAN DEMOCRATIC REPUBLIC
Amtest Associates Ltd.
P.O. Box 55
Addlestone, Surrey KT15 1DU
United Kingdom
Tel: Weybridge (932) 52121
TLX: 928855

GERMAN FEDERAL REPUBLIC
Fluke (Deutschland) GmbH
Krausstrasse 55
8045 Ismaning
West Germany
Tel: (089) 96251
TLX: 0522472
Rapitax: (089) 966718

Fluke (Deutschland) GmbH
Meineckestrasse 53
4000 Dusseldorf 30
West Germany
Tel: (0211)450831
TLX: 8585576

Fluke (Deutschland) GmbH
Oberer Kirchhaldenweg 135
7000 Stuttgart 1
West Germany
Tel: (0711) 694091
TLX: 722518

PK Electronik
Lietzenburger Strasse 91
1000 Berlin 15
West Germany
Tel: (030) 8831058

GREECE
Hellenic Scientific
Representations Ltd
11, Vrasside Street
Athens 615, Greece
Tel: (1) 711140
TLX: 219330

HUNGARY
Amtest Associates Ltd.
P.O. Box 55
Addlestone, Surrey KT15 1DU
United Kingdom
Tel: Weybridge (932) 52121
TLX: 928855

IRAN
Irantronics Company Ltd.
20. Salm Road
Roosevelt Avenue
Tehran, Iran
Tel: 828294/831564
TLX: 212956/212876

IRAQ
Fluke (Holland) B.V.
P.O. Box 5053
Zevenheuvelenweg 53
5004 EB Tilburg, Netherlands
Tel: (013) 673973
TLX: 52237

IRELAND
Euro Electronics
32 Brews Hill
Naven County Meath
Ireland
Tel: (46) 23577
TLX: 3182

ISRAEL
R.D.T. Electronics Engineering Ltd.
P.O. Box 75
46, Sokolov Street
Ramat Hasharon 47235
Israel
Tel: (3) 483211
TLX: 32143

ITALY
Sistrel S.p.A.
Via Giuseppe Armellini # 37
00143 Rome, Italy
Tel: (6) 5915551
TLX: 680356

Sistrel S.p.A.
Via Timavo 66
20099 Sesto S. Giovanni (Milan)
Italy
Tel: (2) 2476693
TLX: 320346

Sistrel S.p.A
Via Cintia
Parco S. Paolo 35
80126 Naples, Italy
Tel: (81) 7679700

KUWAIT
Tareq Company
P.O. Box Safat 20506
Kuwait, Arabian Gulf
Tel: 436100/436045
TLX: 2315

LEBANON AND JORDAN
Mabek (Electronics Division)
P.O. Box 11-3823
Beirut, Lebanon
Tel: 252631/348728
TLX: 22889

LIBYA
Fluke (Holland) B.V.
P.O. Box 5053
Zevenheuvelenweg 53
5004 EB Tilburg, Netherlands
Tel: (013) 673973
TLX: 52237

MOROCCO
Mainvest, Residence Moulay Ismail
Bat. C Boulevard Moulay Slimane
Rabat, Morocco
Tel: (7) 27664
TLX: 31036

NETHERLANDS
Fluke (Nederland) B.V.
P.O. Box 5053
Zevenheuvelenweg 53
5004 EB Tilburg, Netherlands
Tel: (013) 673973
TLX: 52237

NORWAY
Morgenstierne & Co. A/A
Konghellegate 3
P.O. Box 6688, Rodelokka
Oslo 5, Norway
Tel: (2) 356110
TLX: 11719

PDR YEMEN
Fluke (Holland) B.V.
P.O. Box 5053
Zevenheuvelenweg 53
5004 EB Tilburg, Netherlands
Tel: (013) 673973
TLX: 52237

POLAND
Amtest Associates Ltd.
P.O. Box 55
Addlestone, Surrey KT15 1DU
United Kingdom
Tel: Weybridge (932) 52121
TLX: 928855

PORTUGAL
Equipamentos de Laboratorio, Ltda.
P.O. Box 1100
1002 Lisbon, Portugal
Tel: (1) 578936/547512
TLX: 18469

QATER, OMAN & UNITED ARAB EMIRATES
Technology Organisation
P.O. Box 5549
Doha, Qatar
Tel: 321431
TLX: 4581

ROMANIA
Amtest Associates Ltd.
P.O. Box 55
Addlestone, Surrey KT15 1DU
United Kingdom
Tel: Weybridge (932) 52121
TLX: 928855

SAUDI ARABIA
Electronic Equipment
Marketing Co. Ltd.
P.O. Box 3750
Riyadh, Saudi Arabia
Tel: 32761/32700/37023
TLX: 201120

SOMALIA
Fluke (Holland) B.V.
P.O. Box 5053
Zevenheuvelenweg 53
5004 EB Tilburg, Netherlands
Tel: (013) 673973
TLX: 52237

SPAIN
Hispano Electronica S.A.
Piligono Industrial Urtinsa
Apartado de Correos 48
Alcorcon (Madrid)
Spain
Tel: (1) 6194108
TLX: 22404/42634

SWEDEN
Teleinstrument AB
P.O. Box 490
S-162 04 Vallingby 4
Sweden
Tel: (8) 380370
TLX: 11347

SWITZERLAND
Traco Electronic Company Ltd.
I. Jenatschstrasse
Postfach, 8027 Zurich
Switzerland
Tel: (1) 2010711
TXL: 54318

SYRIA
Mabek (Electronics Division)
P.O. Box 4238
Damascus, Syria

TURKEY
Erkman Electronik Aletler
Ticaret Anonim Sirketi
Necatiby Cad 92/3
Karakoy, Istanbul, Turkey
Tel: (1) 5461
TLX: 23353

UNITED KINGDOM
Fluke International Corp.-UK
Colonial Way
Watford, Herts, WD 24 TT
United Kingdom
Tel: (923) 40511

U.S.S.R.
Amtest Associates Ltd.
P.O. Box 55
Addlestone, Surrey KT15 1DU
United Kingdom
Tel: Weybridge (932) 52121
TLX: 928855

YEMEN
Fluke (Holland) B.V.
P.O. Box 5053
Zevenheuvelenweg 53
5004 EB Tilburg, Netherlands
Tel: (013) 673973
TLX: 52237

YUGOSLAVIA
Amtest Associates Ltd.
P.O. Box 55
Addlestone, Surrey KT15 1DU
United Kingdom
Tel: Weybridge (932) 52121
TLX: 928855

Customers in the following countries: Bulgaria, Czechoslovakia, Hungary, Poland, Romania, U.S.S.R., and Yugoslavia, contact: Amtest Associates Ltd., P.O. Box 55, Addlestone, Surrey, KT 15 1DU, England, Tel. (0932) 52121

Section 4A
Manual Change Information

INTRODUCTION

This appendix contains information necessary to backdate the manual to conform with earlier pcb configurations. To identify the configuration of the pcb's used in your instrument, refer to the revision letter (marked in ink) on the component side of each pcb assembly. Table 4A-1 defines the assembly revision levels documented in this manual.

NEWER INSTRUMENTS

As changes and improvements are made to the instrument, the are identified by incrementing the revision letter marked on the affected pcb assembly. These changes are documented on a supplemental change/errata sheet which, when applicable, is inserted at the front of the manual.

OLDER INSTRUMENTS

To backdate this manual to conform with earlier assembly revision levels, perform the changes idicated in Table 4A-1.

CHANGES

There are no backdating changes at this printing. All pcb assemblies are documented at their original revision level.

Table 4A-1. Manual Status and Backdating Information

Ref Or Option No.	Assembly Name	Fluke Part No.	* To adapt manual to earlier rev configurations perform changes in desending order (by no.), ending with change under desired rev letter																					
			–	A	B	C	D	E	F	G	H	J	K	L	M	N	P							
A2	Mother PCB Assy.	421492	–	●	●	●	X																	
A3	CPU Module Assy.	495804	–	●	●	●	X																	
A4	DMA Floppy/Clock Module Assy.	493338	–	●	●	●	●	●	X															
A5	Memory Module Assy.	487090	–	–	–	–	–	●	●	●	X													
A6	Electronic Disk Module Assy.	493254	–	●	●	X																		
A7	Video/Keyboard Module Assy.	495770	–	●	●	●	●	●	●	●	●	X												
A8	IEEE-488 Interface Module Assy.	497420	–	–	–	●	●	●	●	X														
A9	Power Supply Module Assy.	493320	–																					
A9A1	P/S Control PCB Assy.	493429	–	●	●	●	●	●	●	●	●	●	●	X										
A9A2	P/S Rectifier PCB Assy.	493437	–	●	●	●	●	●	X															
A9A3	P/S Switching Transistor PCB Assy.	497826	–	●	●	●	X																	

* X = The PCB revision levels documented in this manual.
● = These revision letters were never used in the instrument.
–= No revision letter on the PCB.

Section 5
List of Replaceable Parts

TABLE OF CONTENTS

ASSEMBLY NAME	DRAWING NO.	PAGE
Final Assembly	(1720A-T&B/5001)	5-3
A1 Bezel Assembly	(1720A-4202)	5-3
A1A1 Front Panel PCB Assembly	(1720A-4008)	5-3
A2 Mother PCB Assembly	(1720A-4002)	5-3
A3 CPU Module Assembly	(1720A-7604K)	5-3
A4 DMA Floppy/Clock Module Assembly	(1720A-7614K)	5-3
A5 Memory Module Assembly	(1720A-7603K)	5-3
A6 Electronic Disk (Spare)	(1720A-4017/-001)	5-4
A7 Video/Keyboard Module Assembly	(1720A-7609K)	5-4
A8 IEEE-488 Interface Module Assembly	(1720A-7613K)	5-4
A9 Power Supply Module	(1720A-7630K)	5-4
A9A1 Power Supply Control PCB Assembly	(1720A-4006)	5-4
A9A2 Power Supply Rectifier Assembly	(1720A-4007)	5-5
A9A3 Transister Switching Assembly	(1720A-4016)	5-5
A10 Floppy Disk Replacement Module	(1720A-7635K)	5-5
Field Service Kits		5-5
Accessories		5-5

5-1. INTRODUCTION

5-2. This section contains parts breakdown of the instrument. Components are listed alphanumerically by assembly.

5-3. Parts lists include the following information:

1. Reference Designation.

2. Description of each part.

3. FLUKE Stock Number.

4. Federal Supply Code for Manufactures. (See Section 4 for Code-to-Name list.)

5. Manufacture's Part Number.

6. Total Quantity of components per assembly.

7. Recommended Quantity: This entry indicates the recommended number of spare parts necessary to support one to five instruments for a period of 2 years. This list presumes an availability of common electronic parts at the maintenance site. For maintenance for 1 year or more at an isolated site, it is recommended that at least one of each assembly in the instrument be stocked (see paragraph 5-7). In the case of optional subassemblies, plug-ins, etc., that are not always part of the instrument, or are deviations from the basic instrument model, the REC QTY column lists the recommended spares quantity for the items in that particular assembly.

5-4. HOW TO OBTAIN PARTS

5-5. Components may be ordered directly from the manufacturer by using the manufacturer's part number, or from the John Fluke Mfg. Co., Inc. or it authorized representative by using the FLUKE STOCK NUMBER.

NOTE

Module assemblies must be ordered with the FLUKE STOCK NUMBER. Do not use the manufacturer's part number to order these assemblies.

In the event the part you order has been replaced by a new or improved part, the replacement will be accompanied by an explanatory note and installation instructions, if necessary.

5-6. To ensure prompt and efficient handling of your order, include the following information.

1. Quantity.

2. FLUKE Stock Number.

3. Description.

4. Reference Designation.

5. Printed Circuit Board Part Number and Revision Letter.

6. Instrument Model and Serial Number.

5-7. A Recommended Spare Parts Kit for your basic instrument is available from the factory. This kit contains those items listed in the REC QTY column of the parts list in the quantities recommended.

5-8. Parts price information is available from the John Fluke Mfg. Co., Inc. or its representative. Prices are also available in a Fluke Replacement Parts Catalog, which is available on request.

CAUTION

Indicated devices are subject to damage by static discharge.

Final Assembly 1720A Instrument Controller

REF DES	DESCRIPTION	FLUKE STOCK NO.	MFG SPLY CODE	MFG PART NO. OR TYPE	TOT QTY	REC QTY	NOTE
	1720A INSTRUMENT CONTROLLER (1720A-T&B/5001) THIS PARTS LIST IS AT PCB OR ASSEMBLY LEVELS, RECOMMENDED PARTS ARE LISTED FOLLOWING EACH ASSEMBLY.						
MP1	FLOPPY POWER CABLE	503003	89536	503003	1	1	
MP2	FOOT	544957	89536	544957	1	1	
MP3	CORD SET, 6A, 250V, 7-1/2 FT.	284174	89536	284174	1	1	
MP4	CORNER DECAL	296269	89536	296269	4	1	
MP5	DISKETTE 10-PACK	Y1706	89536	Y1706	1	1	
MP6	DIAGNOSTIC DISKETTE	521419	89536	521419	1	1	
MP7	SYSTEM DISKETTE	521401	89536	521401	1	1	
MP8	PROGRAMMER KEYBOARD	Y1720	89536	Y1720	1	1	
TM1	OPERATOR MANUAL	518647	89536	518647	1	1	
TM2	USER MANUAL	518654	89536	518654	1	1	
TM3	BASIC PROGRAMMING MANUAL	518670	89536	518670	1	1	
TM4	SERVICE MANUAL	518662	89536	518662	1	1	
TM5	PROGRAMMING WORK SHEET	533547	89536	533547	1	1	
TM6	BASIC PROGRAMMING REF GUIDE	526210	89536	526210	1	1	
TM7	HPL/BASIC MANUAL	546341	89536	546341	1	1	
A1	BEZEL ASSEMBLY (1720A-4202T)	SEE	NOTE	*	1		
	CRT KIT (CRT & ELECTRONICS) (1720A-7640K)	577379	89536	537472	1	1	
	TOUCH SENSE PANEL	483644	89536	483644	1	1	
A1A1	FRONT PANEL PCB ASSEMBLY (1720A-4008)	493445	89536	493445	1	1	
A2	MOTHER PCB ASSEMBLY (1720A-4002T)	421492	89536	421492	1	1	
A3	CPU MODULE ASSEMBLY (1720A-7604K)	577320	89536	495804	1	1	
	HDT PROM, EVEN BYTE	576322	89536	576322	1	1	
	HDT PROM, ODD BYTE	576314	89536	576314	1	1	
	STANDARD PROM 2708 ODD BYTE	495796	89536	495796	1	1	
	STANDARD PROM 2708 EVEN BYTE	495788	89536	495788	1	1	
	(NOTE: ODD BYTE PROM MOUNTS CLOSEST TO MICROPROCESSOR.)						
	IC, N-CHAN, MICROPROCESSOR	473249	01295	TMS9900	1	1	
A4	DMA FLOPPY/CLOCK MODULE ASSEMBLY (1720A-7614K)	577361	89536	493338	1	1	
U12	IC, DIG. CLOCK CHIP, 4-CTR	507756	07263	FCM6020	1	1	
U15	IC, DMA CONTROLLER	504316	52840	DM1883B	1	1	
U16	IC, FLOPPY DISK CONTROLLER	504308	52840	FD1791B	1	1	
Y1	CRYSTAL, 32.768 HZ	501817	51791	CX-1V-32.768HZ	1	1	
A5	MEMORY MODULE ASSEMBLY (1720A-7603K)	577312	89536	487090	1	1	
U19 THRU U27 U33 THRU U41 U46 THRU U54 U59 THRU U67	IC, MEMORY	483487	01295	TMS4116-25NL	36	7	
U57	IC, DIGITAL	483529	34649	D3242	1	1	

Final Assembly 1720A Instrument Controller (cont)

REF DES	DESCRIPTION	FLUKE STOCK NO.	MFG SPLY CODE	MFG PART NO. OR TYPE	TOT QTY	REC QTY	NOTE
A6	ELECTRONIC DISC (SPARE) (1720A-4017T/-001)	518472	89536	OPTION 001 **	AR	1	
S1	SWITCH, DIP, 4-POS	408559	10582	206-4	1	1	
U4	IC, ADDRESS MUX & REFRSH CTR	483529	34649	D3242	1	1	
U10, U13 U18 THRU U21 U26 THRU U29 U34 THRU U37 U42 THRU U45 U50 THRU U53 U58 THRU U61 U66 THRU U69 U73 THRU U76	IC, RAM MEMORY	523654	50088	MK4332D-2	36	7	
Y1	CRYSTAL, 10 MHZ, 10 PPM	385732	09579	183025B	1	1	
A7	VIDEO/KEYBOARD MODULE ASSEMBLY (1720A-7609K)	577346	89536	495770	1	1	
J24	PHONO JACK, RIGHT ANGLE	500819	11503	901	1	1	
U10,U28	IC, DIGITAL	483552	01295	TMS9902NL	2	1	
U36	IC, DIG., GENERATOR	534776	04713	MCM66740	1	1	
U41	IC, DIGITAL	483503	34649	C8275	1	1	
U42	IC, DIGITAL	504332	01295	TMS9981N	1	1	
U47	PROGRAM ROM	535032	89536	535032	1	1	
S2	SWITCH, SLIDE	452862	89536	452862	1	1	
Y1	CRYSTAL, 10 MHZ	520239	89536	520239	1	1	
A8	IEEE-488 INTERFACE MODULE ASSEMBLY (1720A-7613K)	577353	89536	497420	1	1	
J1	RECEPT, CONNECTOR	485763	00779	552791-2	1	1	
J2	RECEPT, CONNECTOR	500884	00779	552791-1	1	1	
MP1	BRACKET, CONN. SUPPORT	545616	89536	545616	1	1	
MP2	BRACKET, GROUND	545624	89536	545624	1	1	
S1,S2	SWITCH ASSY., DIL, 8-POS, SPDT	404490	00779	435166-5	2	1	
S3	SWITCH, SLIDE (BLACK)	393629	10389	23-021-114	1	1	
A9	POWER SUPPLY MODULE INCLUDES A9A1, A9A2, A9A3 (1720A-7630K)	577338	89536	493320	1	1	
B1	BATTERY, 4 AMP HR., 6V	501379	56309	PS640	1	1	
MP1	AIR FILTER	524660	89536	524660	1	1	
A9A1	POWER SUPPLY CONTROL PCB ASSEMBLY (1720A-4006)	493429	89536	493429	1	1	
F1	FUSE, SLO-BLO, 3 AMP, 250V	109280	71400	MDA3	1	5	
MP1	INSTRUMENT FOOT	544957	89536	544957	2	2	
MP2	LINE FILTER	500405	89536	500405	1	1	
MP3	FAN ASSEMBLY	501312	89536	501312	1	1	
MP4	CAP, (TO XF1 FUSEHOLDER)	460238	89536	460238	1	1	
XF1	FUSE HOLDER W/NUT (F1)	460329	89536	460329	1	1	
S1	SWITCH, ROCKER, 3-POLE (CODE 3)	513838	73559	T1HL50-1L-BL-FNBL	**	1	
S1	SWITCH, ROCKER, 2-POLE (CODE 4)	542720	73559	T1GL50-1L-BL-FNBL	AR	1	

Final Assembly 1720A Instrument Controller (cont)

REF DES	DESCRIPTION	FLUKE STOCK NO.	MFG SPLY CODE	MFG PART NO. OR TYPE	TOT QTY	REC QTY	NOTE
A9A2	POWER SUPPLY RECTIFIER ASSEMBLY (1720A-4007)	493437	89536	493437	1	1	
CR14	DIODE, RECTIFIER, 3 AMP	524231	04713	MBR-320M	1	1	
F1	FUSE, 5 AMP, 250V	109215	71400	MDA5	1	5	
K1	RELAY	512574	82227	28D111ZE-0231	1	1	
MP1	BUTTON, POWER SWITCH	545814	89536	545814	1	1	
Q1,Q2	XSTR, SI, NPN	418459	01295	TIP-31	2	1	
S1	SWITCH, SLIDE	453365	79727	G1-116-0005,G-20-32	1	1	
U4	IC, LIN, NEG. VOLT REGULATOR	394551	04713	MC7905CP	1	1	
A9A3	TRANSISTOR SWITCHING ASSEMBLY (1720A-4016)	SEE	NOTE	*	1	1	
Q1,Q2	XSTR, SI, NPN, POWER	495705	04713	MJ10007	2	1	
Q3,Q4	XSTR, SI	495697	04713	MPS-U06	2	1	
Q5,Q6	XSTR, SI	495689	04713	MPS-U56	2	1	
A10	FLOPPY DISK REPLACEMENT MODULE (1720A-7635K)	521393	89536	500488	1	1	
	FIELD SERVICE KITS						
	#1 CRT KIT	577379	89536	537472 **	AR	1	
	#3 SERVICE KIT (MODULE)	537506	89536	537506 **	AR	1	
	70% LEVEL KIT (PARTIAL)	539361	89536	539361 **	AR	1	
	ACCESSORIES						
	1 EXTENDER CARD	Y1704	89536	Y1704 **	AR	1	
	2 NULL MODEM ADAPTER	Y1705	89536	Y1705 **	AR	1	
	3 2M RS-232 CABLE	Y1707	89536	Y1707 **	AR	1	
	4 10M RS-232 CABLE	Y1708	89536	Y1708 **	AR	1	
	5 2M RS-232 PRINTER CABLE	Y1709	89536	Y1709	AR	1	
	6 SHIPPING CASE	Y1711	89536	Y1711 **	AR	1	
	7 RACK MOUNT KIT	Y1790	89536	Y1790 **	AR	1	
	8 MAINTENANCE KIT	577791	89536	577791	AR	1	

* NON-PROCURABLE AT ASSEMBLY LEVEL

** AR = CUSTOMER OPTION

Section 6
Schematics

TABLE OF CONTENTS

FIGURE	TITLE	PAGE
6-1	List of Mnemonics and Bus Signal Definitions	6-2
6-2	1720A Top, Bottom and Side Views	6-5
6-3	1720A Interconnection Diagram	6-6
6-4	CPU Module Assembly	6-7
6-5	Memory Module Assembly	6-13
6-6	IEEE-488 Interface Module Assembly	6-19
6-7	Video Keyboard Module Assembly	6-26
6-8	DMA Floppy/Clock Module Assembly	6-31
6-9	Power Supply Module Assembly	6-38
6-10	Electronic Disk Module Assembly	6-42
6-11	Keyboard Schematic	6-48
6-12	Mother Board Schematic	6-49
6-13	Front Panel Schematic	6-50

Bus Signals				
NAME	**MNEMONIC**	**NUMBER OF LINES**	**TYPE OF DRIVER**	**ACTIVE LEVEL**
Address Bus	A00-A19	20	Tri-State Buffer	High
Data Bus	DI00-DI15	16	Tri-State Buffer	High
CRU Bus				
CRU Input	CRIN	1	Open Collector	High
CRU Output	CROUT	1	TTL Buffer	High
CRU Clock	CRCLK-	1	TTL Buffer	Low
CRU Address Acknowledge	CRACK-	1	Open Collector	Low
Control Bus				
Address Valid	ACVAL-	1	Tri-State Buffer	Low
Address Acknowledge	ADACK-	1	Open Collector	Low
Bus Write	BUSWR-	1	Tri-State Buffer	Low
Refresh-Interrupt	RINT-	1	Tri-State Buffer	Low
Bus Request In	RQIN-	1	TTL	Low
Bus Request Out	RQOUT	1	TTL	Low
Bus Grant In	GRIN	1	TTL	Low
Bus Grant Out	GROUT-	1	TTL	Low
Clock Sync	SYNC	1	TTL Buffer	N/A
Byte Mode	BMODE	1	TTL	Low
Hardware Error	HERR-	1	Open Collector	Low
Bus Clear	BCLR-	1	TTL Buffer	Low
ACOK	ACOK	1	TTL Buffer	High
DCOK	DCOK	1	TTL Buffer	High
Test	TEST-	1	TTL Buffer	Low
Halt	HALT-	1	Open Collector	Low
Power Supply Lines				
+5 Volts	+5V	2		
+12 Volts	+12V	2		
Logic Common	LCOM	8		
-12 Volts	-12V	1		
Battery Backed Up Supplies				
+5 Volts	+5VB	1		
-5 Volts	-5VB	1		
+12 Volts	+12VB	1		

Bus Signal Definitions	
SIGNAL NAME	**DEFINITION**
ADDRESS BUS	(A00-A19) - 20 lines used by the processor and Direct memory Access (DMA) devices to address a word of memory or an I/0 Register. The lower 12 bits (A00-A11) are also used for Communications Register Unit (CRU) addressing.
DATA BUS	(DI00-DI15) - 16 lines for data transfer throughout the system. In addition, when the Refresh-Interrupt control line is asserted, a device requests interrupt service by driving its assigned data bus line to the low (logic 0) state. Interrupts are prioritized with a request on data bit 0 (DI00) having the highest priority and data bit 15 (DI14) the lowest. Only one interrupting device is allowed per line.
ADDRESS VALID	(ADVAL-) - Indicates the presence of valid address information on the address bus. Also indicates that data on the DATA/INTERRUPT lines is true during processor write operations.

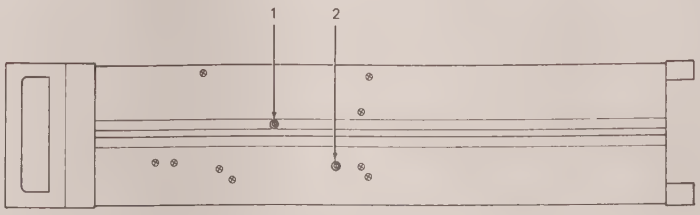
Figure 6-1. List of Mnemonics and Bus Signal Definitions

SIGNAL NAME	DEFINITION
ADDRESS ACKNOWLEDGE	(ADACK-) - Response line from an addressed memory or I/0 device. Must be asserted to indicate address recognition and allow processor (or DMA device) to procees with I/0 or memory operation.
BUS WRITE	(BUSWR-) - Indicates direction of data transfer on DATA/INTERRUPT lines. A low level indicates that processor (or DMA device) is sending data to the address I/0 or memory register.
REFRESH-INTERRUPT	(RRINT-) When asserted, indicates that interrupting devices may request interrupt service by driving the appropriate data line low. Dynamic memory devices use this line to initiate a refresh cycle.
BUS REQUEST IN	(RQIN-) - Active low request by device which wishes to control the bys (typically a high speed I/0 device). This line changes at each connector. It is called Bus Request Out (RQOUT-) as it leaves the connector, having passed through the device installed in that connector. From this connector, the line goes to the next higher priority device (closer to processor) where it will again be called Bus Request In (the processor has only a RQIN- line). The device which is physically closest to the processor has the highest request priority. Each device inhibits transfer of the RQIN signal to the RQOUT line if and only if it is also requesting the bus and no lower priority device has been granted the bus, signified by a low logic level on Bus Grant In (GRIN-).
BUS REQUEST OUT	(RQOUT-) - Active low signal to signify that a device is requesting control of the bus. This line will be asserted by the device to request bus usage or if the Bus Request In line is asserted. A device may not request the use of the bys if it has already been granted to a lower priority device signified by a low level on Bus Grant In. See also Bus Request In. A non-DMA device should connect Bus Request In to Bus Request Out to maintain continuity through the device.
BUS GRANT IN	(GRIN-)-Active low indication that the processor has places its tri-state bus drivers in the high impedance mode and entered the hold state. This line is broken at each connector and is called Bus Grant Out (GROUT-) as it leaves each device enroute to the next lower priority devie (further from processor), where it is called Bus Grant In.
BUS GRANT OUT	(GROUT-) - Each device normally transfers the logic state of the Bus Grant In line to the Bus Grant Out line. A device inhibits this transfer only when it is requesting or has been granted control of the bus. At this time, the Bus Grant Out line is maintained at a logic high level. Non-DMA devices shouldconnect Bus Grant In to Bus Grant Out to maintain continuity of the grant signal.
CLOCK SYNC	(SYNC) - Six megahertz clock.
BUS CRU INPUT	(CRIN) - communications Register Unit (CRU) input line, sampled during execution of the Store Communications Register (STCR) and Test Bit (TB) instructions.
BUS CRU OUTPUT	(CROUT) - CRU output line. Serial data is placed on this line during execution of the Load Communication Register (LDCR), Set Bit to Zero (SBZ), and Set Bit to One (SBO) instructions. Data is valid when the CRU clock line is low.
CRU ADDRESS ACKNOWLEDGE	(CRCLK-) - Active low indication that valid data appears on the CROUT line. (CRACK-) - Active low response line from addressed CRU devices. Must be asserted by CRU device when its address is placed on the address bus.
HARDWARE ERROR	(HERR-) - Active low line indicates that a device has detected a fatal error, such as a memory parity error, during a read or refresh operation. Assertion of this line will initiate a processor load sequence immediately following the instruction being executed.
BUS CLEAR	(BCLR-) - A low level on this line indicates that the bus interface circuitry on all devices should be reset to the unaddressed and inactive state. BDLR- is active for 160 ns.
ACOK	(AC OK) - A high level indicates that the unit is receiving adequate ac power. A transition from the high to low state indicates that the unit may soon lose dc power and backup operations should commence.

Figure 6-1. List of Mnemonics and Bus Signal Definitions (cont.)

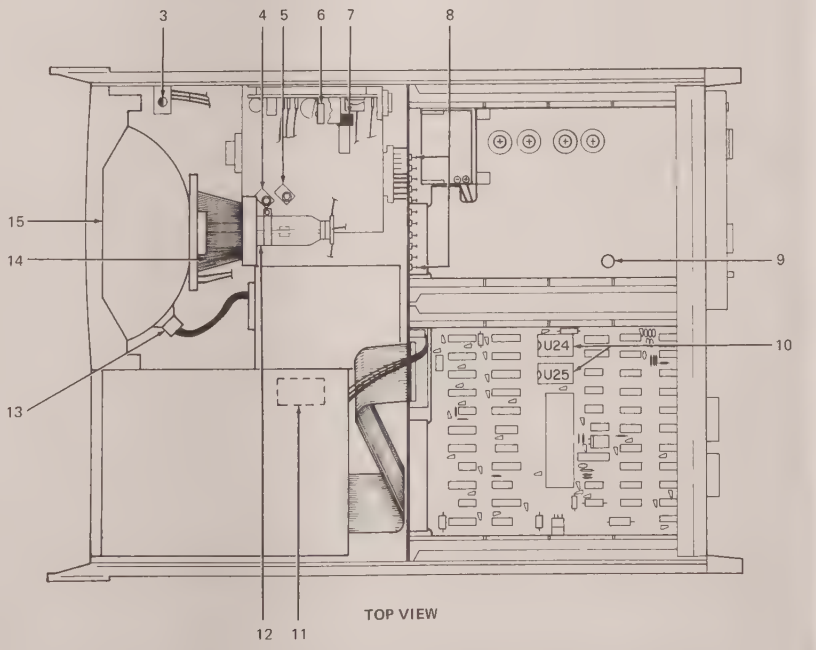
SIGNAL NAME	DEFINITION
BYTE MODE	(BMODE) - A low level indicates that the bus master is expecting data in byte format. DI08 through DI15 contain valid data. DI00 through DI07 are not used.
DCOK	(DC OK) - This line is intended for use as a power on/off reset. When this line as low all operation in the unit umst cease since the power supplies are close to falling out of regulation. The only exception will be circuitry capable of rumming off the battery supply (BATT).
TEST	(TEST-) - This signal enables those devices with special diagnostic circuits to enter a special mode of operation.
HALT	(HALT-) - This active low line will initiate a load sequence.

Figure 6-1. List of Mnemonics and Bus Signal Definitions (cont.)

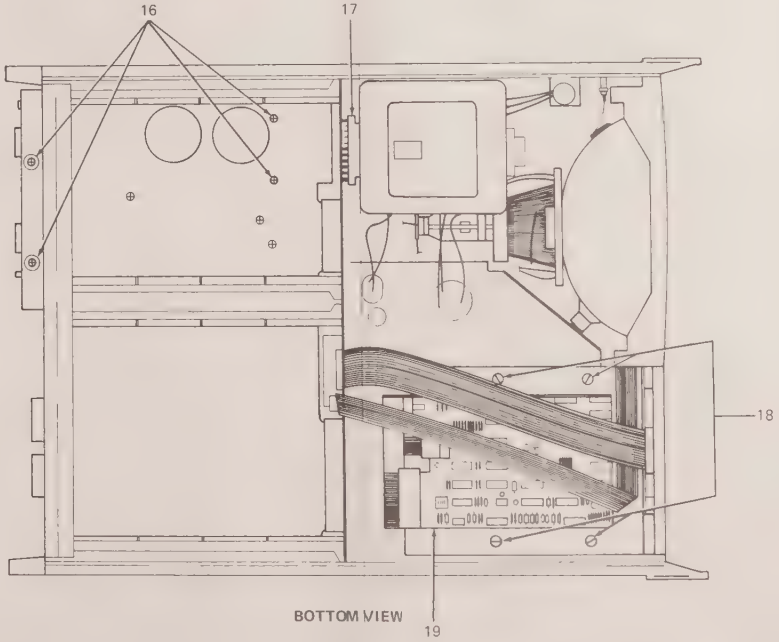


SIDE VIEW

ITEM NO.	NAME	ITEM NO.	NAME
1	Horizontal Osc Coil (L50)	11	Disk Drive Power Connector (on PCB)
2	Horizontal Size Coil (L53)	12	Yoke centering rings
3	Raster Pot on the chassis bracket	13	CRT HV Lead
4	Vertical Linearity Pot	14	CRT Yoke
5	Vertical Size	15	CRT
6	Focus Pot	16	Four screws that hold the Power Supply PCB to the rest of the Power Supply.
7	Master Brightness Pot in the CRT Electronics Area (on vertical PCB).	17	CRT Electronics Assembly Edge Connector
8	TP1 through TP11 (labeled on Mother Board)	18	Four screws that pass through the bottom mounting plate into the Floppy Disk Drive Unit Cover.
9	5V Adjust Pot	19	Floppy Disk Drive PCB
10	PROMs		



TOP VIEW



BOTTOM VIEW

Figure 6-2. Top, Bottom and Side Views

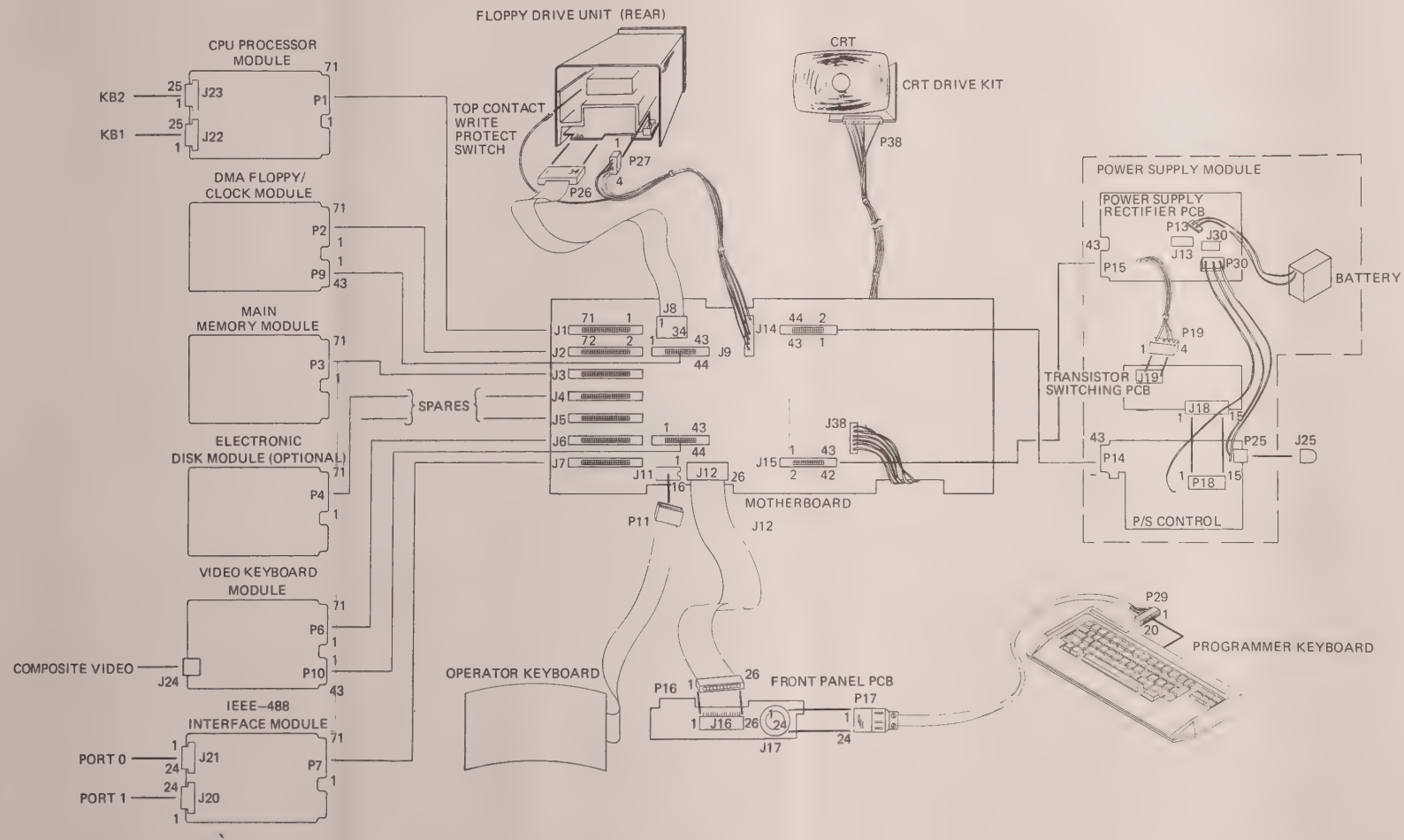


Figure 6-3. Interconnection Diagram

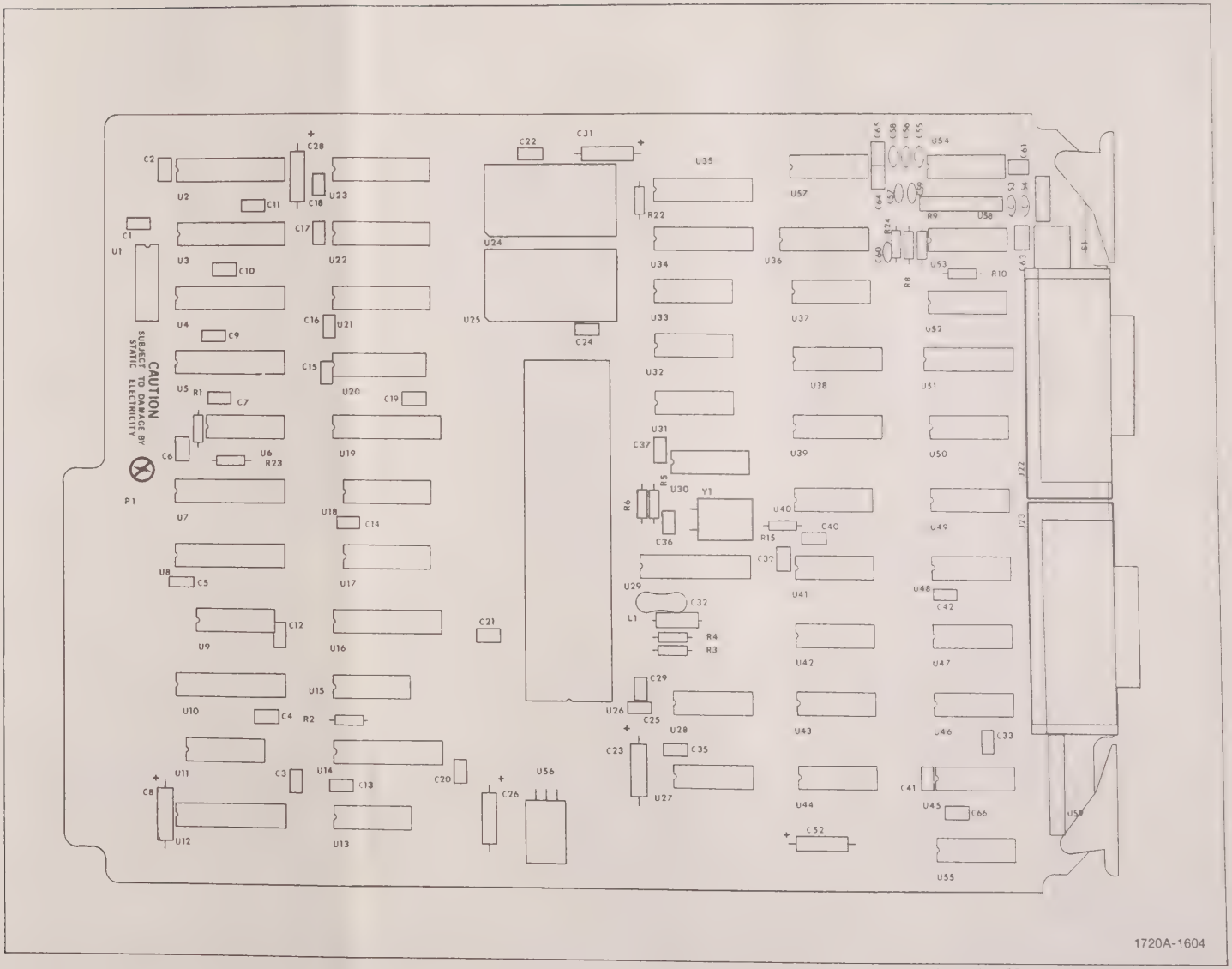
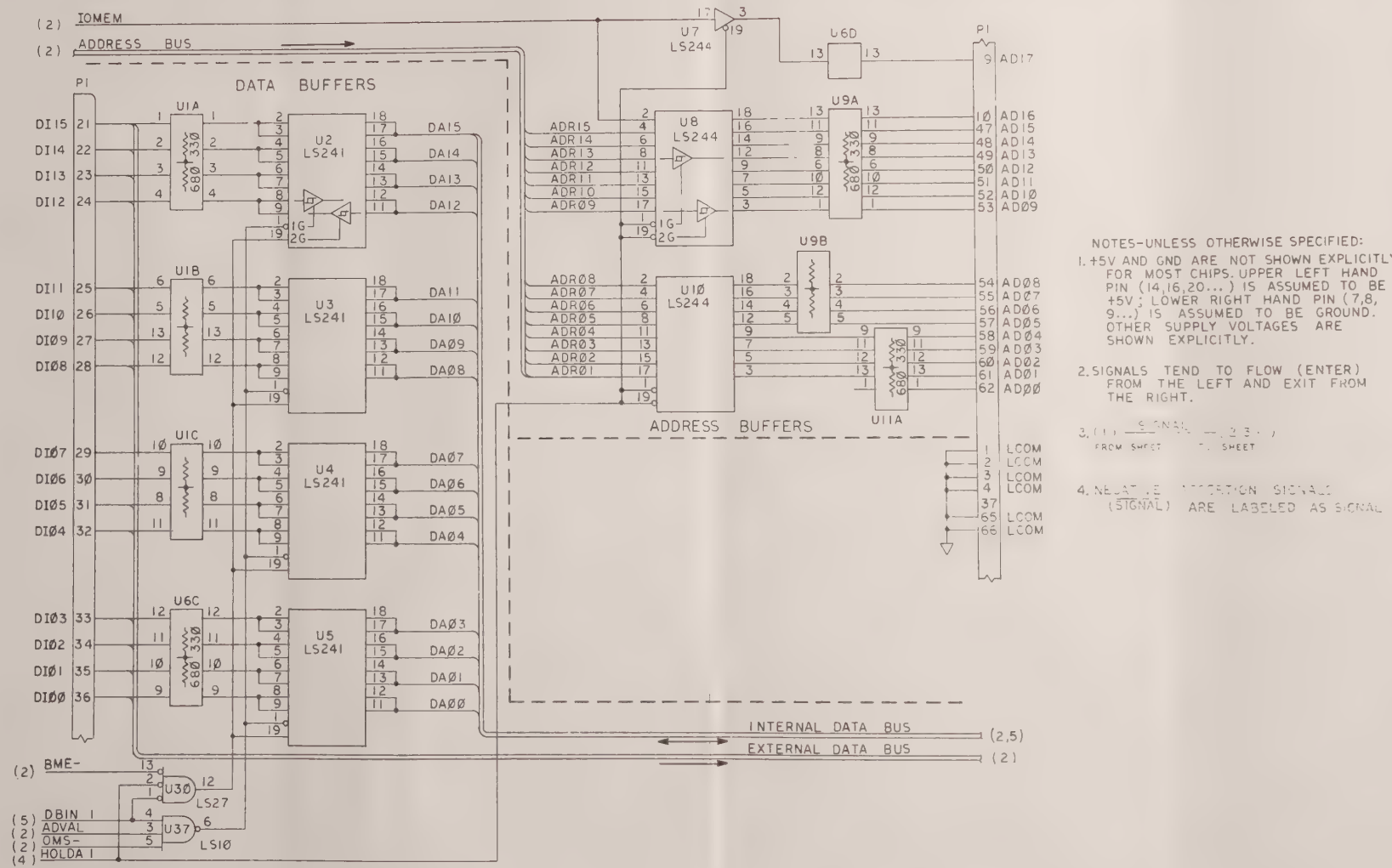
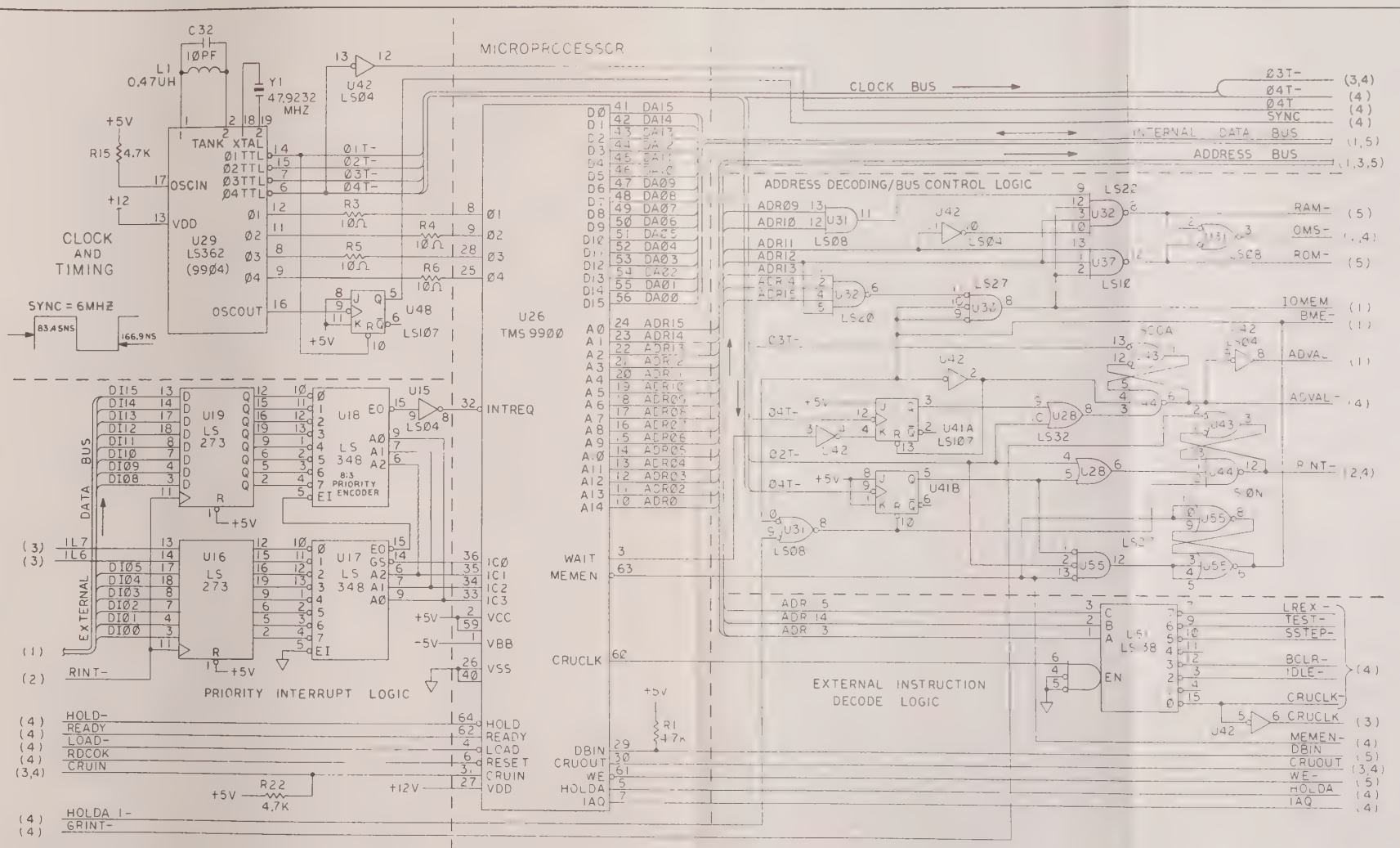


Figure 6-4. CPU Module Assembly



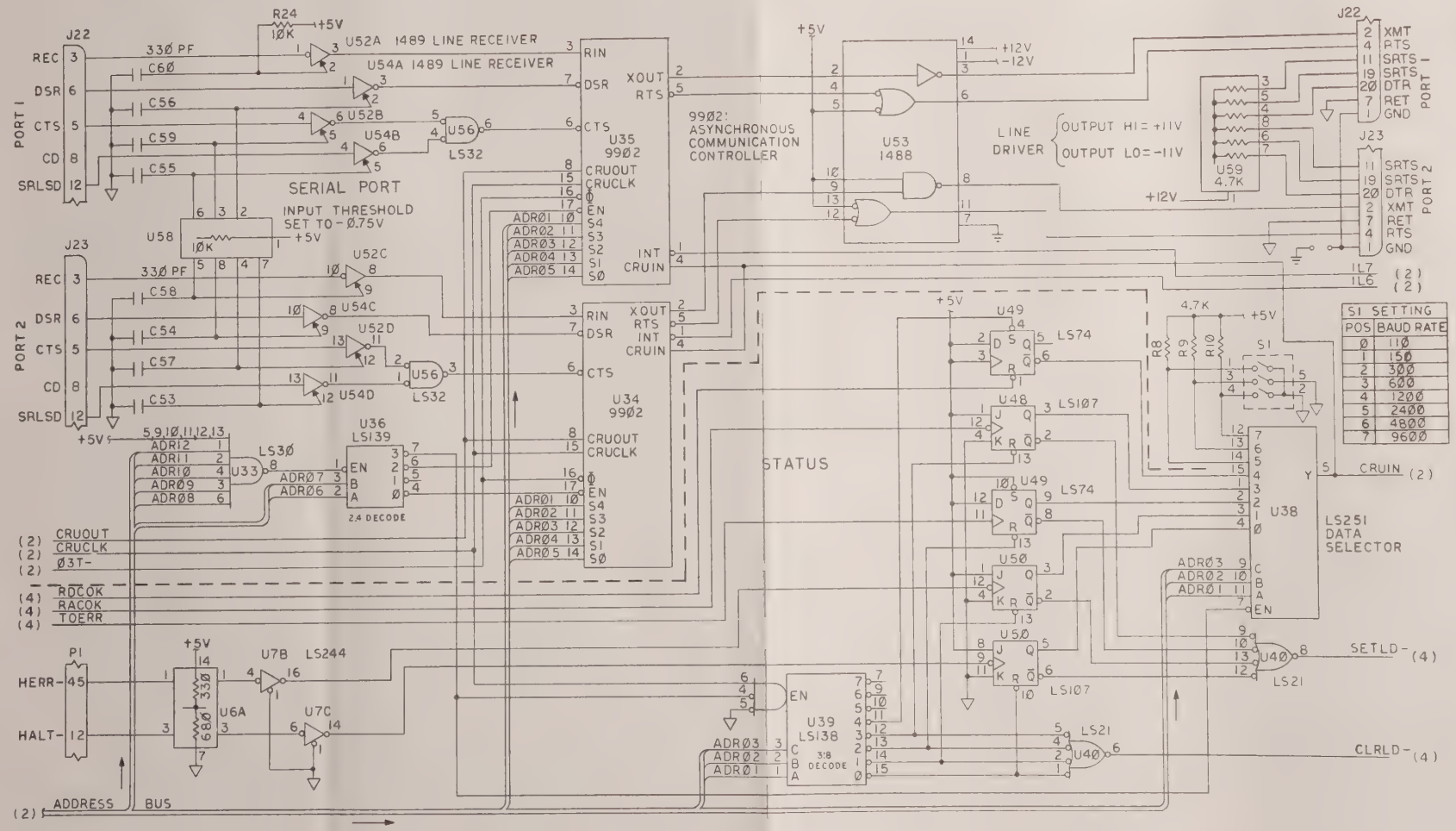
1720A-1004
(Sheet 1 of 5)

Figure 6-4. CPU Module Assembly (cont.)



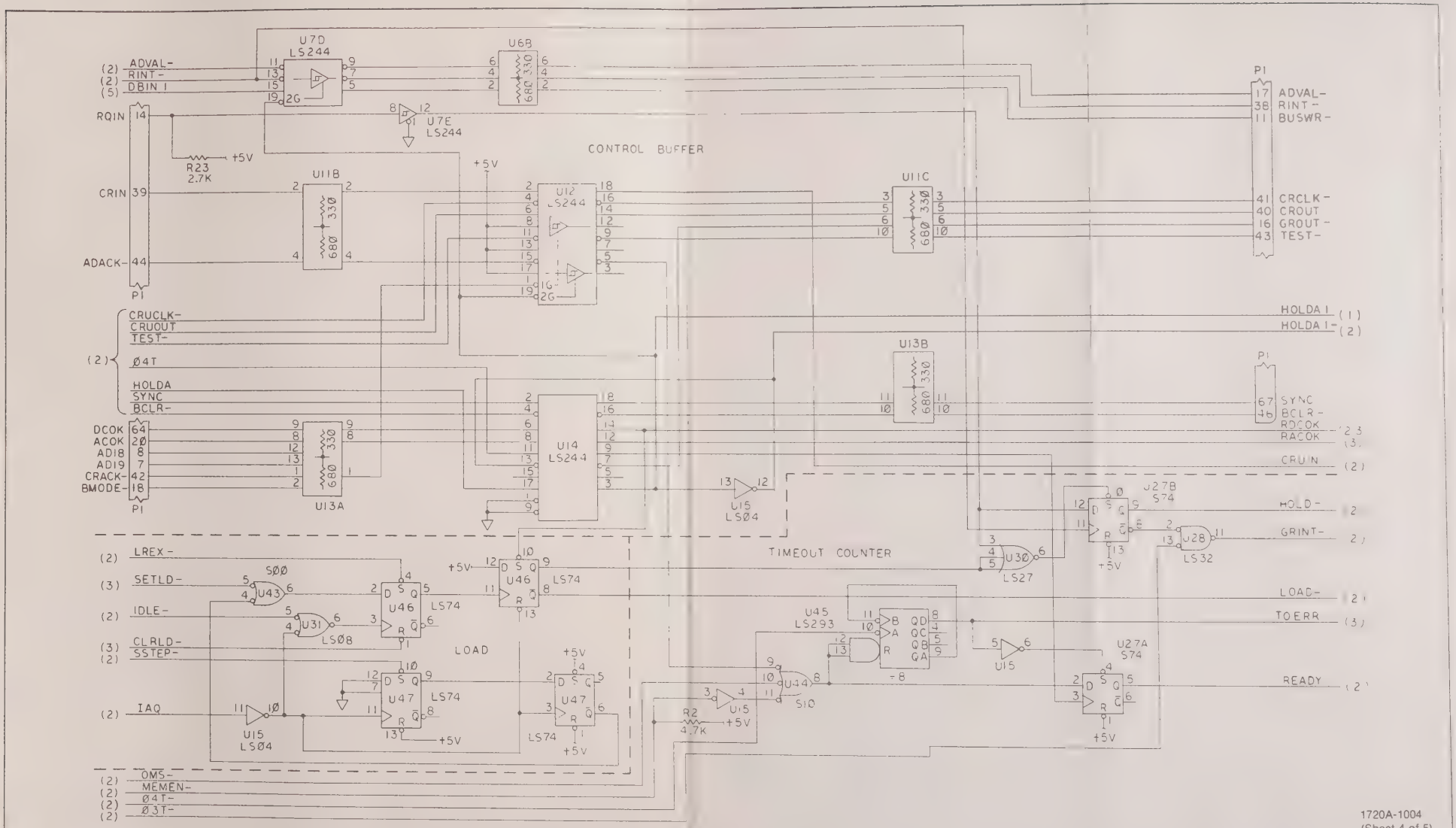
1720A-1004
(Sheet 2 of 5)

Figure 6-4. CPU Module Assembly (cont.)



1720A-1004
(Sheet 3 of 5)

Figure 6-4. CPU Module Assembly (cont.)



1720A-1004
(Sheet 4 of 5)

Figure 6-4. CPU Module Assembly (cont.)

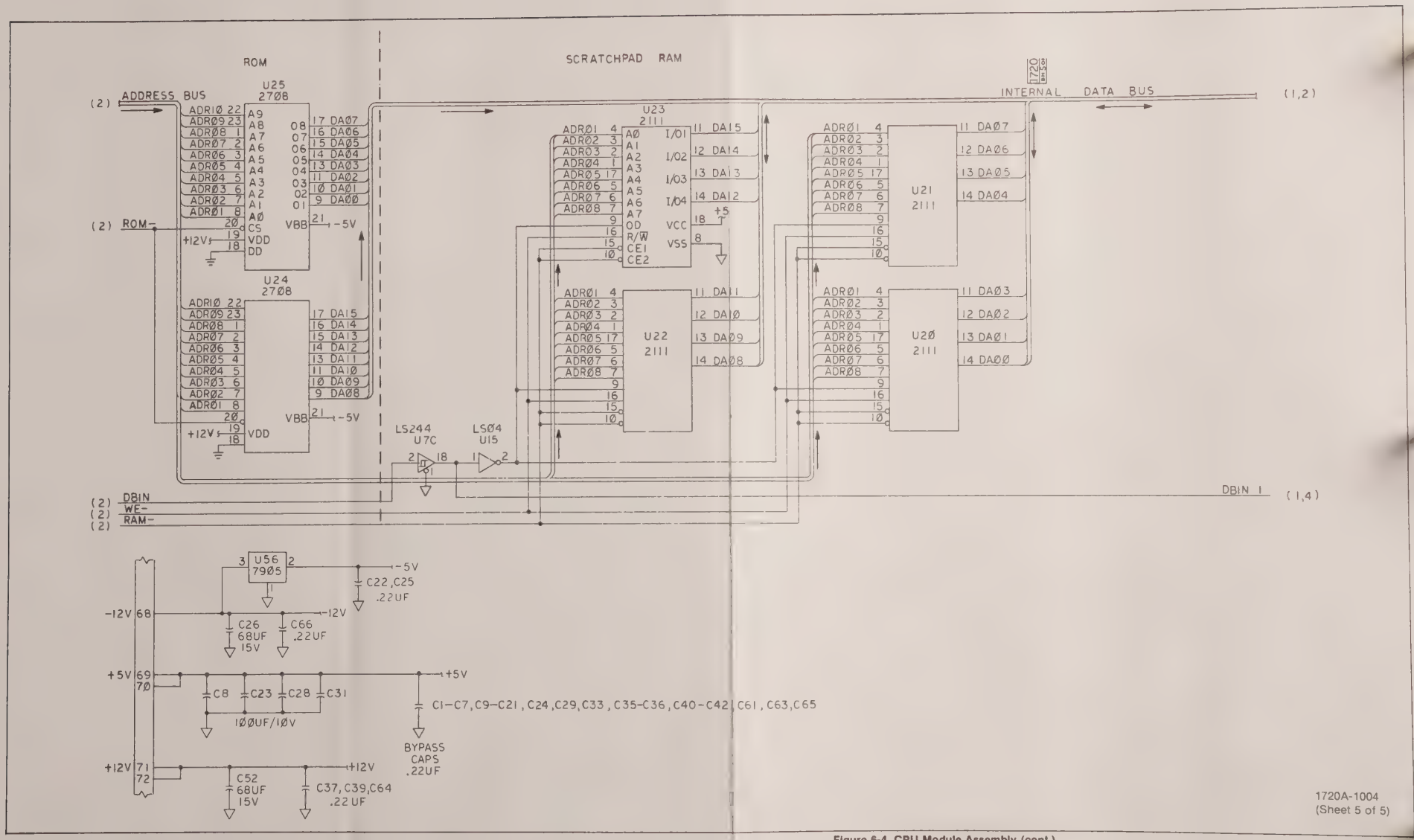
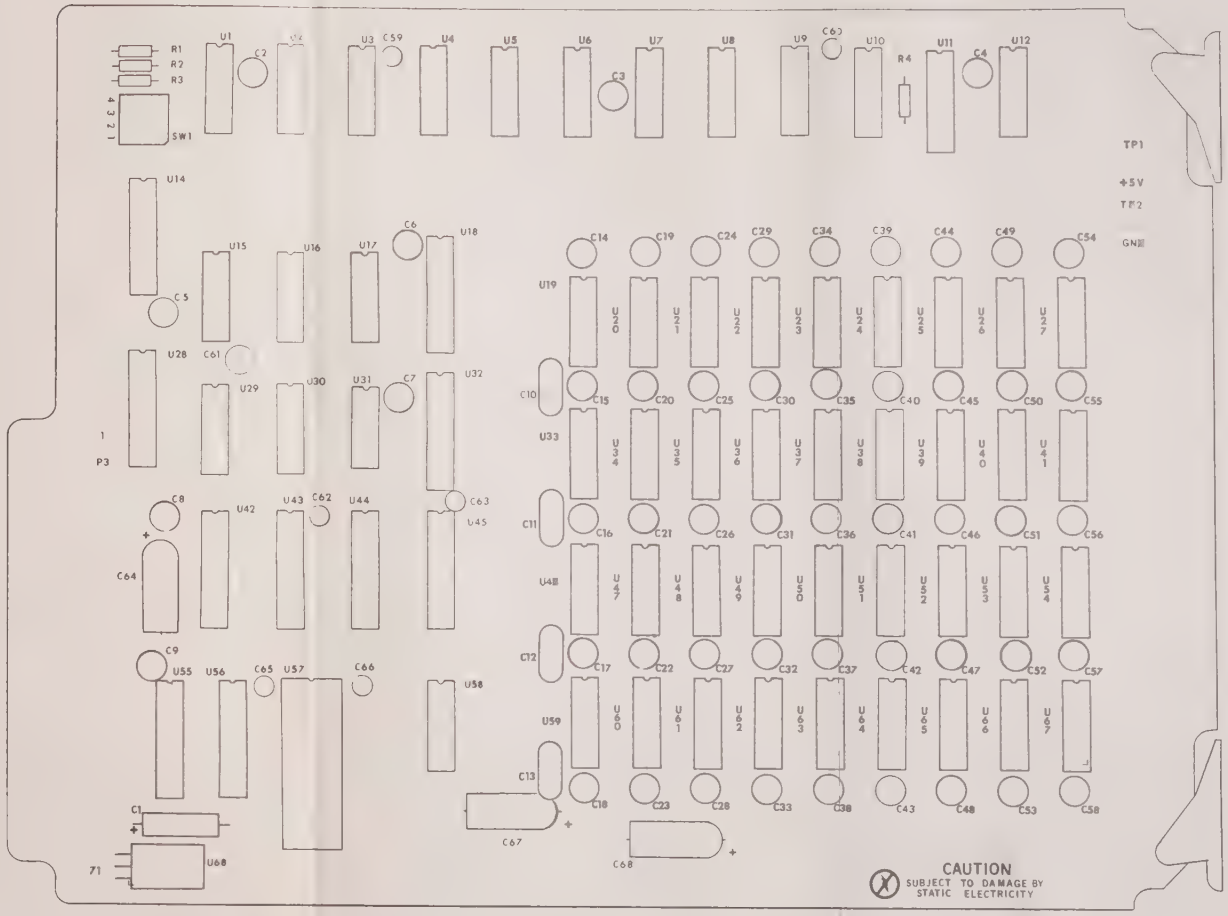


Figure 6-4. CPU Module Assembly (cont.)



1720A-1603

Figure 6-5. Memory Module Assembly

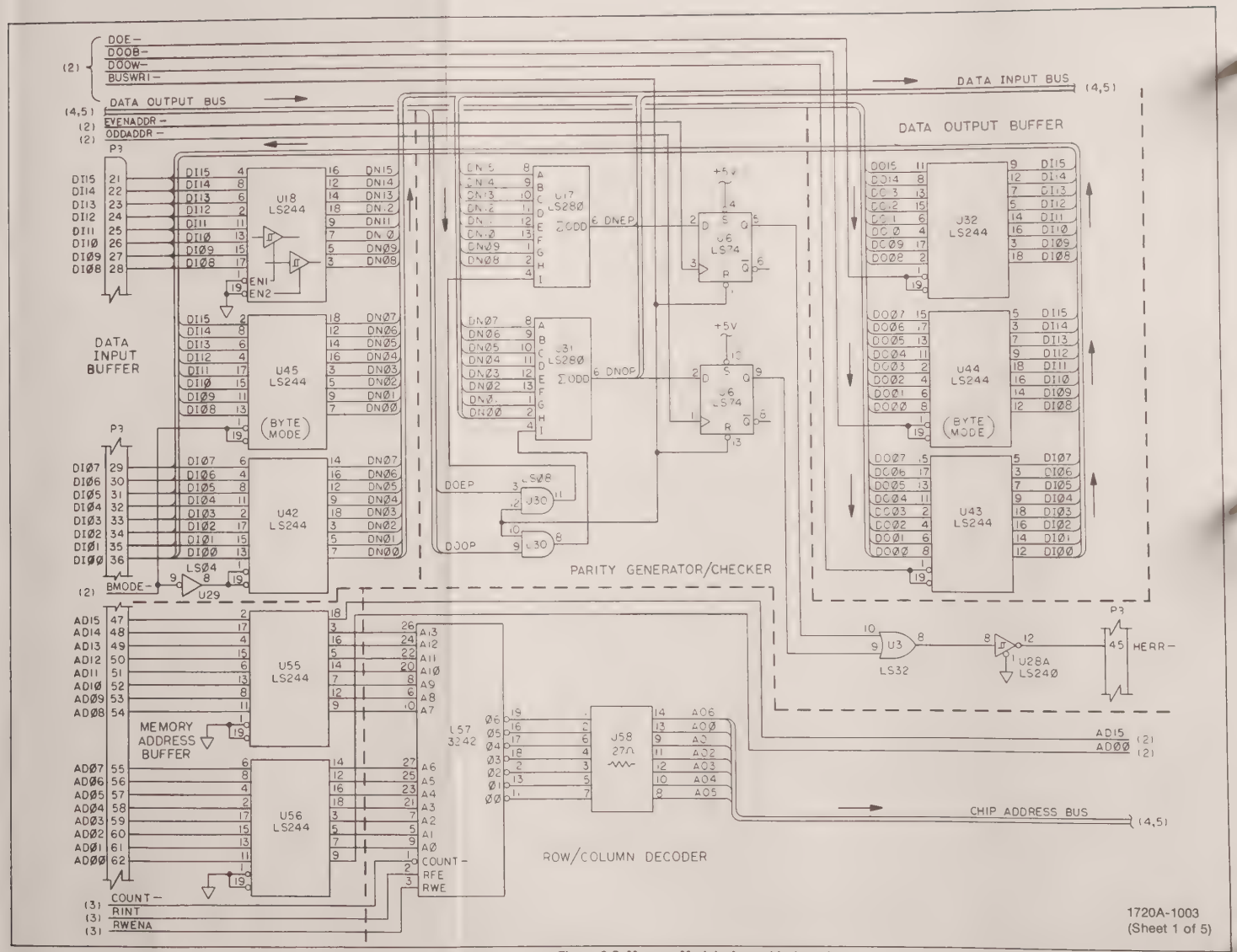
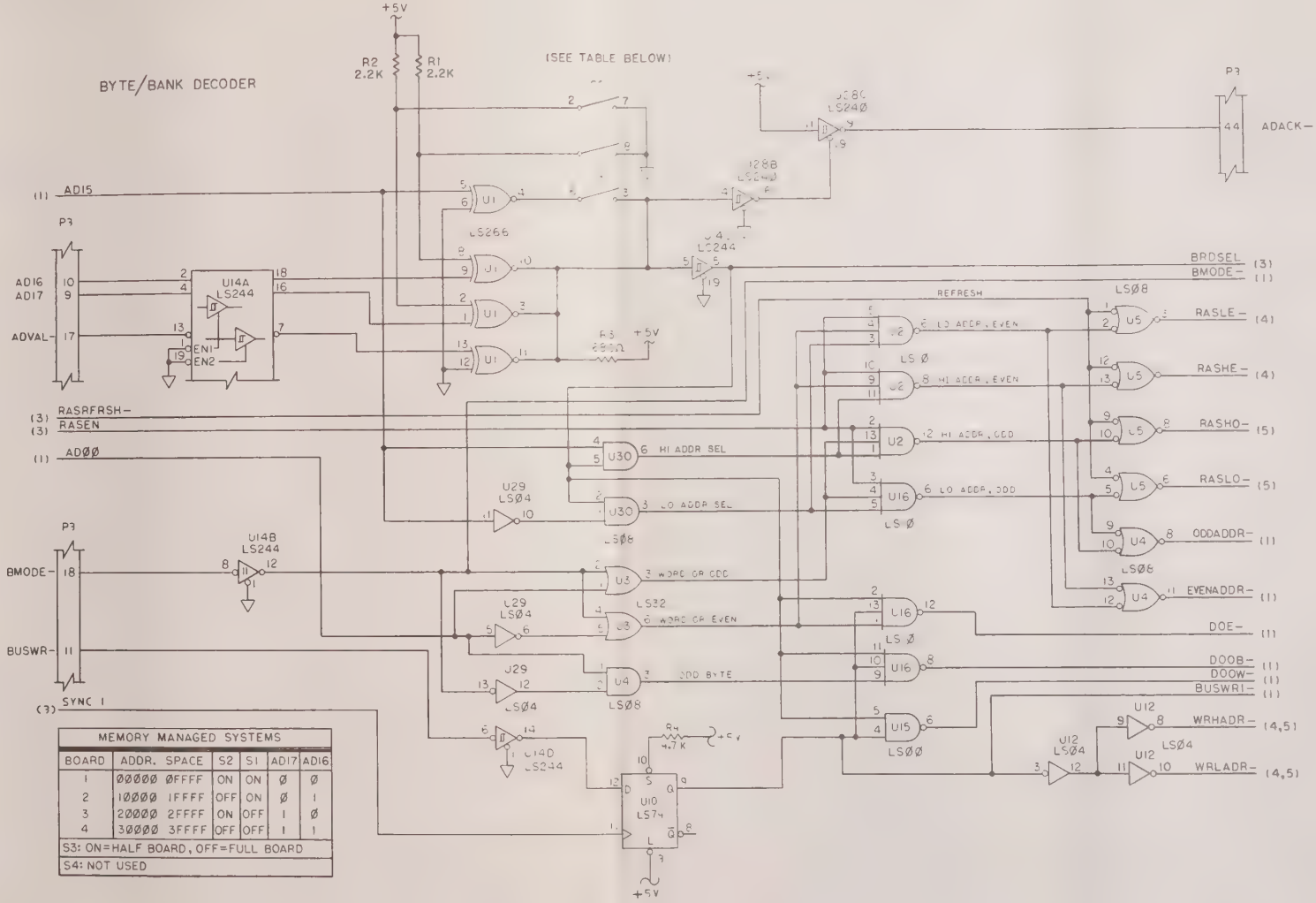


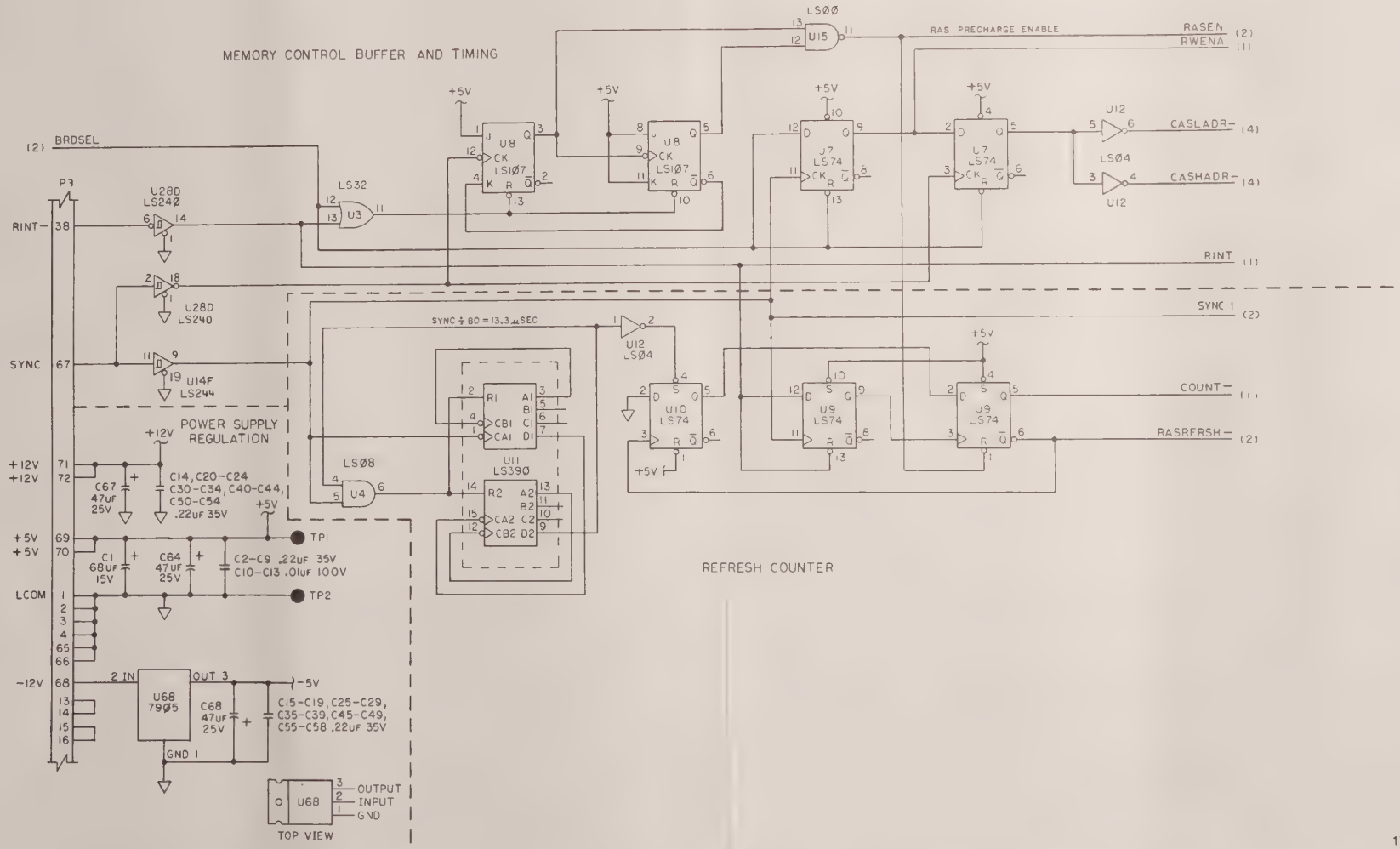
Figure 6-5. Memory Module Assembly (cont.)



MEMORY MANAGED SYSTEMS					
BOARD	ADDR. SPACE	S2	S1	AD17	AD16
1	00000 0FFFF	ON	ON	0	0
2	10000 1FFFF	OFF	ON	0	1
3	20000 2FFFF	ON	OFF	1	0
4	30000 3FFFF	OFF	OFF	1	1
S3: ON=HALF BOARD, OFF=FULL BOARD					
S4: NOT USED					

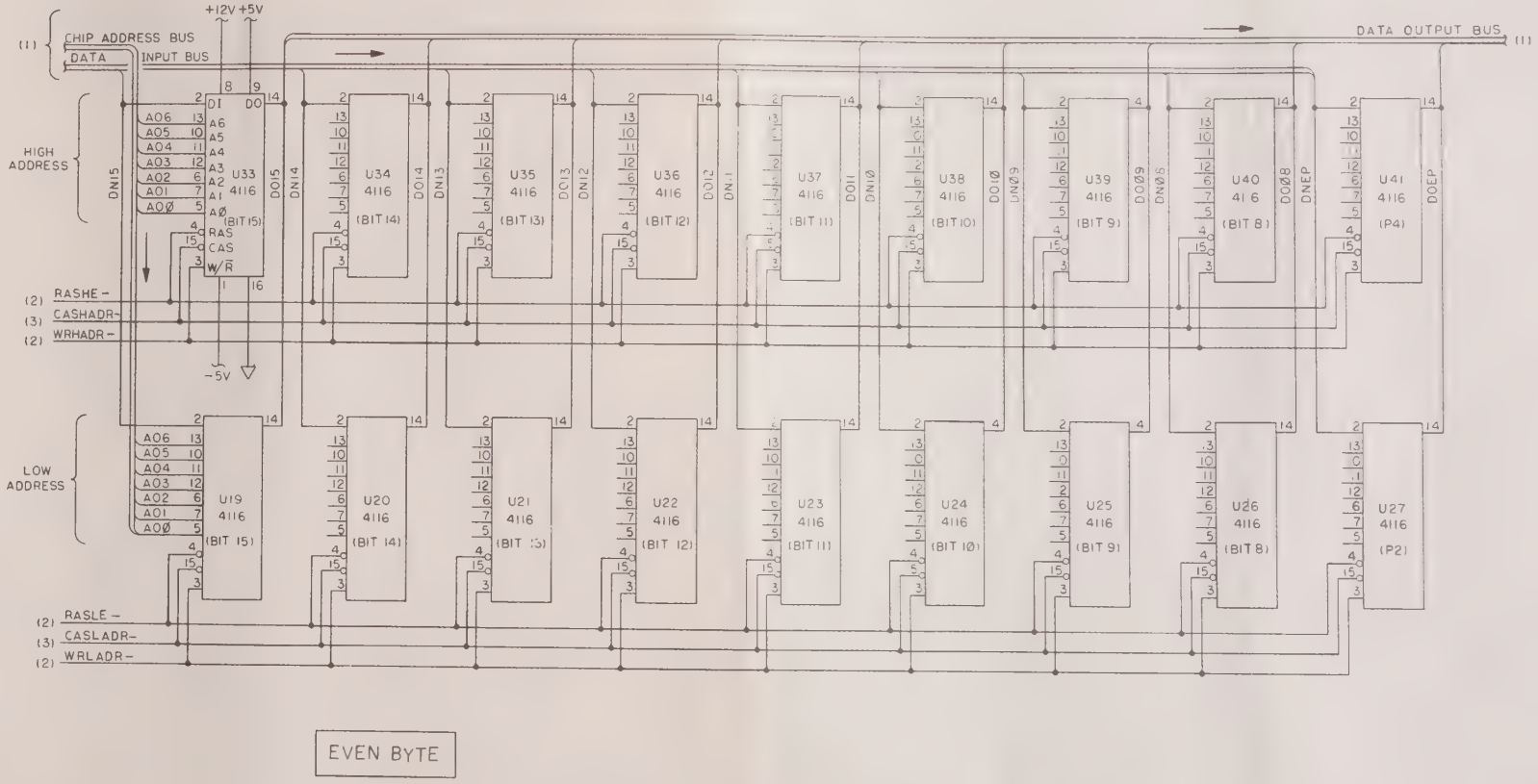
1720A-1003
(Sheet 2 of 5)

Figure 6-5. Memory Module Assembly (cont.)



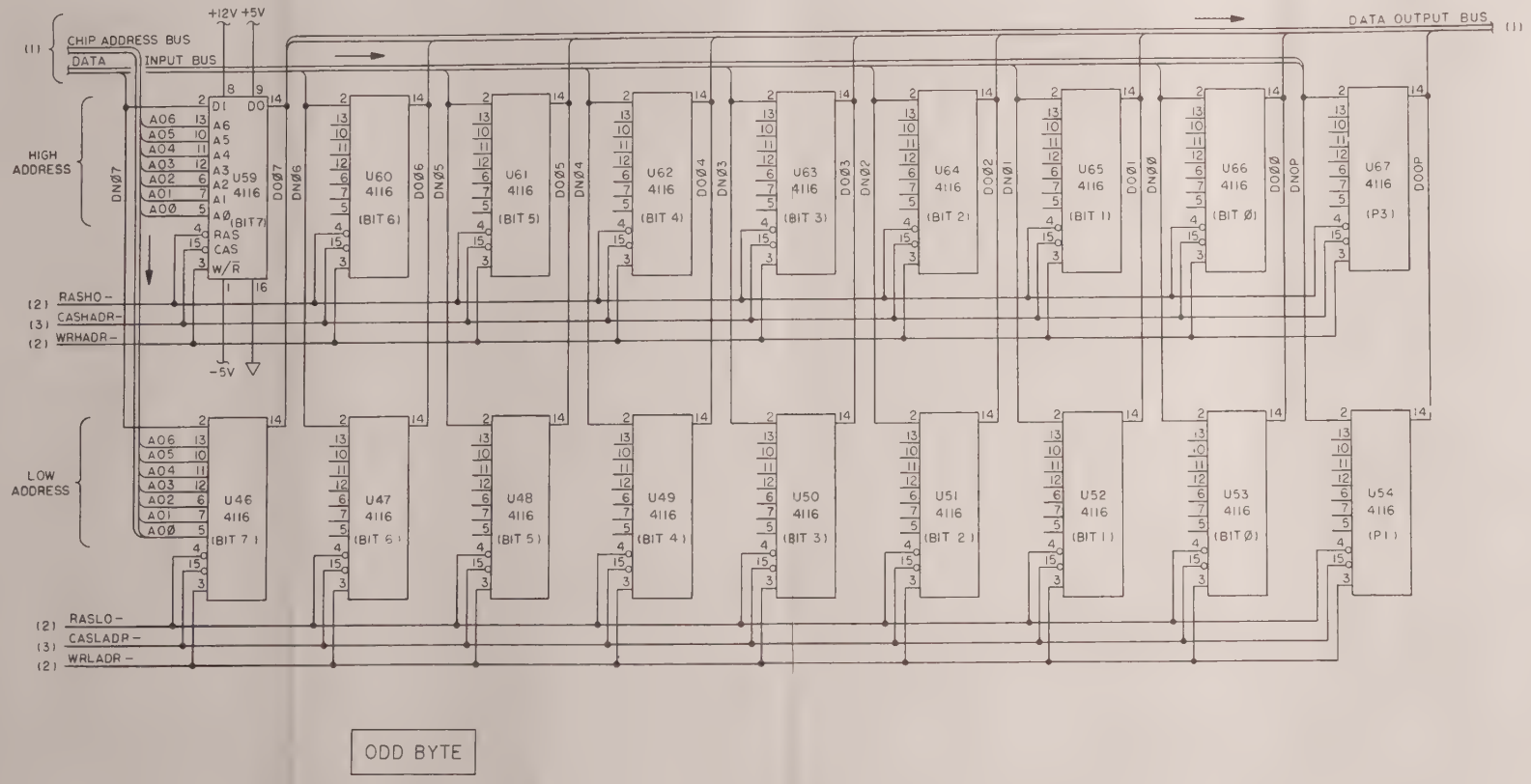
1720A-1003
(Sheet 3 of 5)

Figure 6-5. Memory Module Assembly (cont.)



1720A-1003
(Sheet 4 of 5)

Figure 6-5. Memory Module Assembly (cont.)



1720A-1003
(Sheet 5 of 5)

Figure 6-5. Memory Module Assembly (cont.)

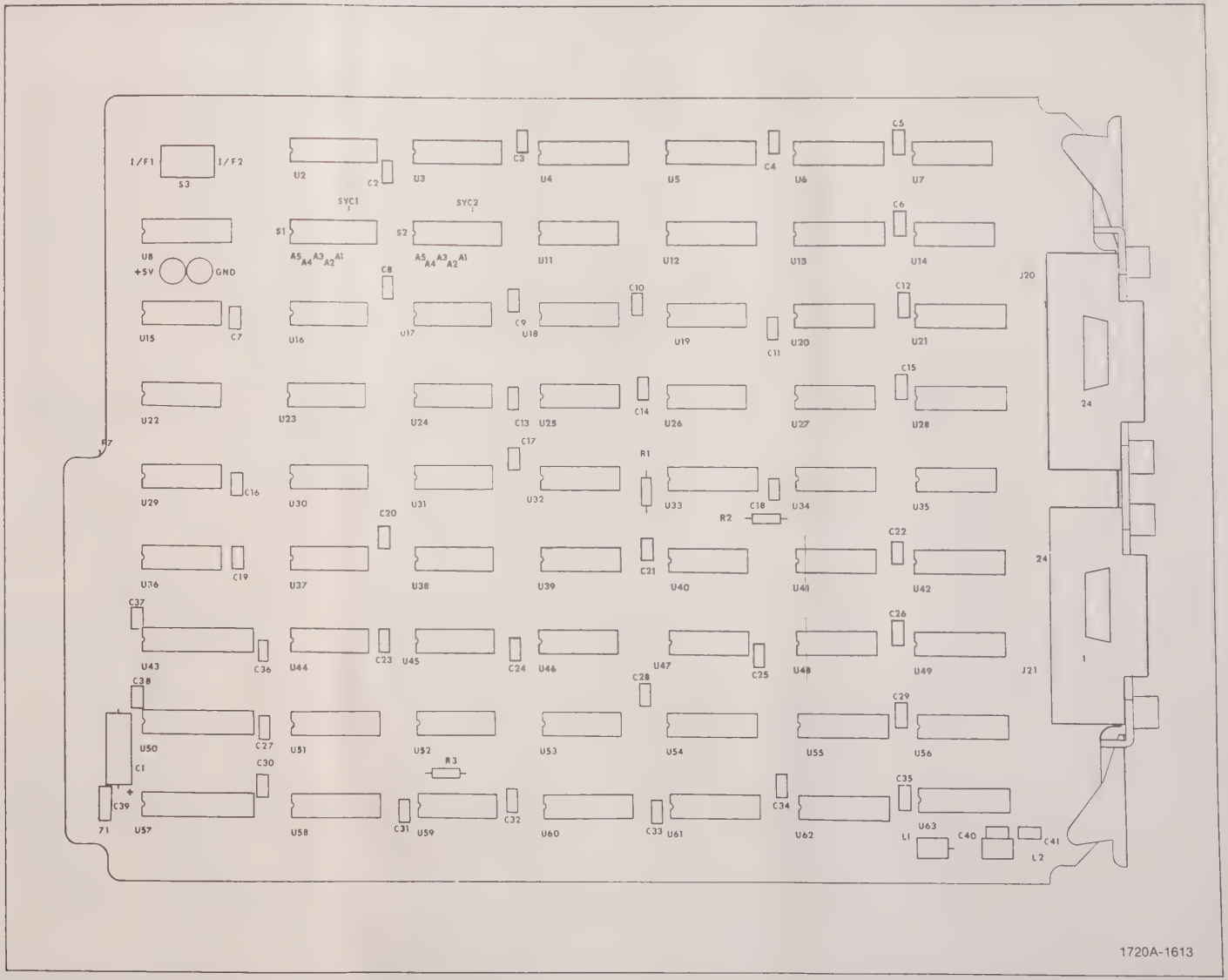
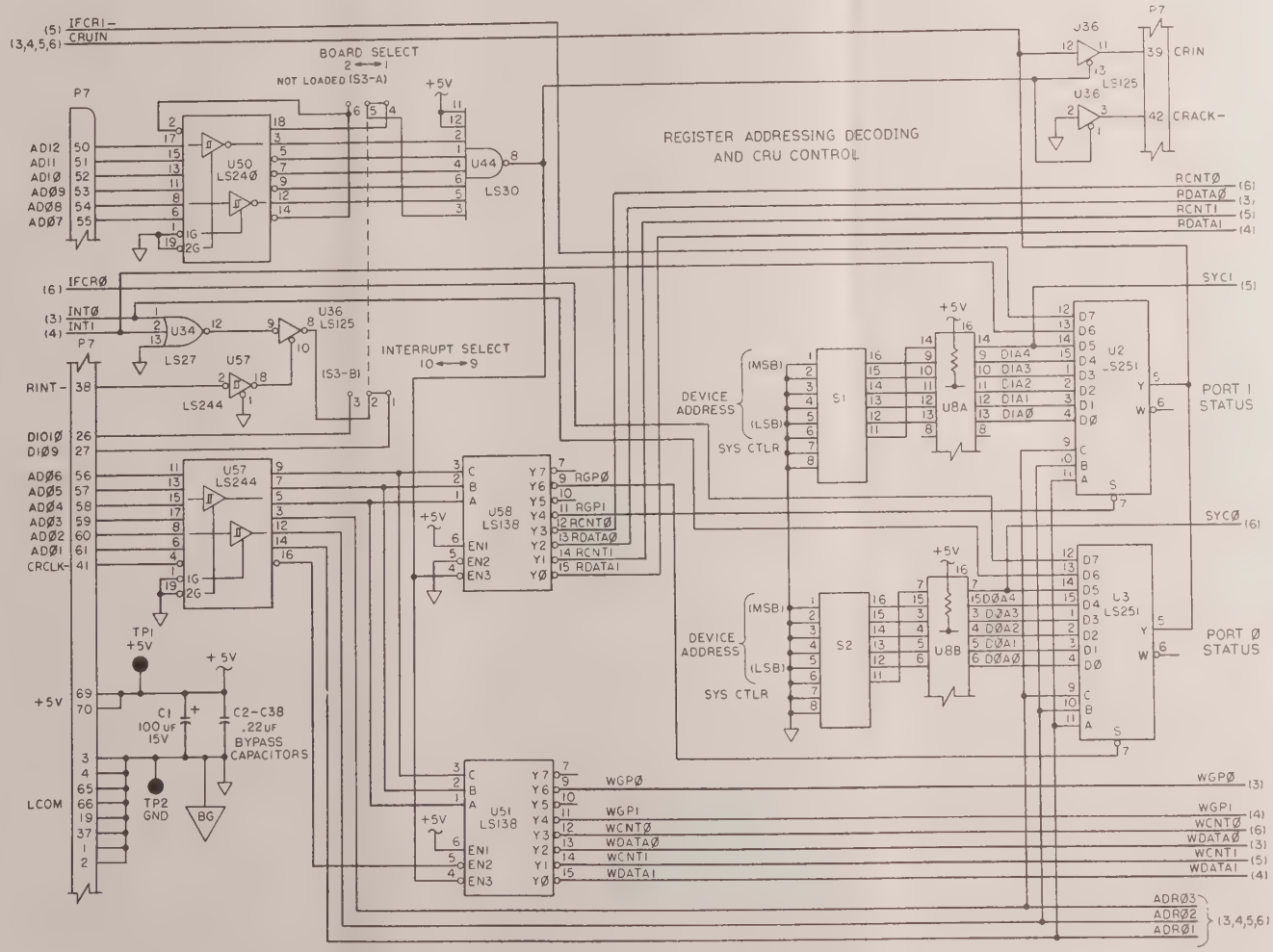


Figure 6-6. IEEE-488 Interface Module Assembly

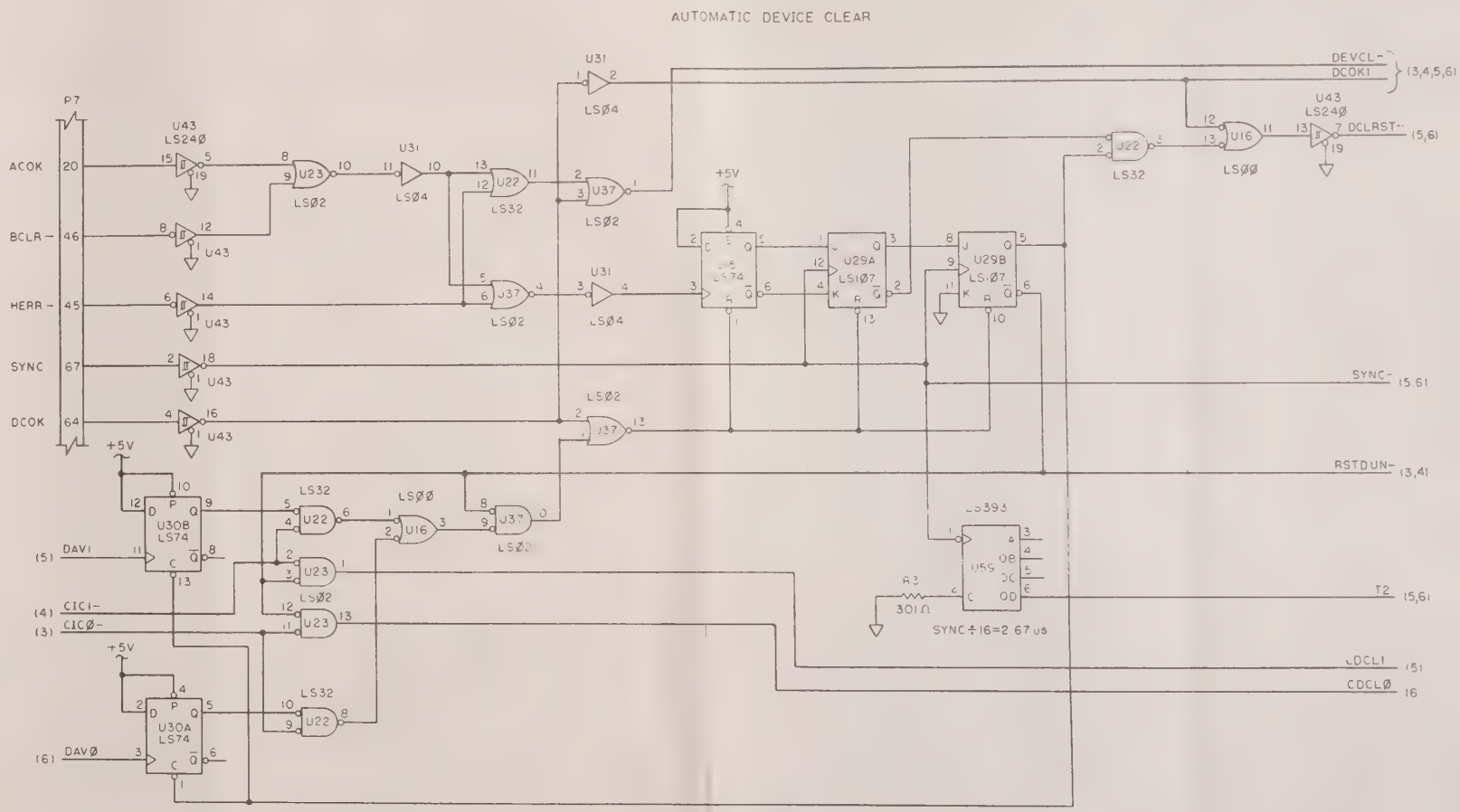


NOTES – UNLESS OTHERWISE SPECIFIED.

1. ALL GPIB (IEEE 488) SIGNALS ON J2Ø AND J2I ARE LOW TRUE ASSERTION (TTL LEVELS); I.E. A ZERO VOLT READING INDICATES A LOGICAL ONE.
2. BYPASS CAPACITORS ARE DISTRIBUTED 1 PER 3446 AND BY EVERY OTHER REMAINING PACK.
3. CONNECT BG LINE AT THE CONNECTOR P7, 100 MIL TRACE.

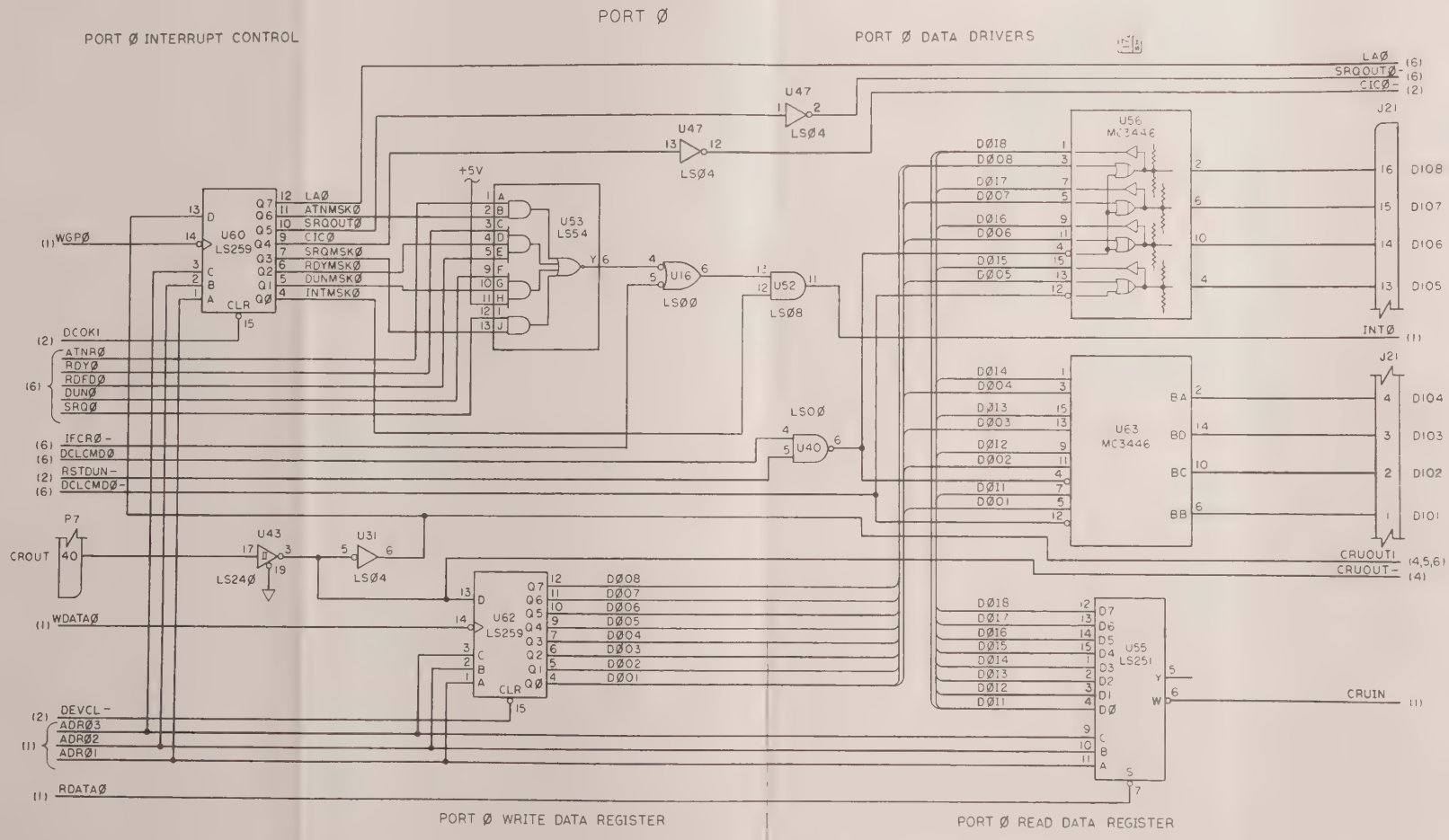
1720A-1013
(Sheet 1 of 6)

Figure 6-6. IEEE-488 Interface Module Assembly (cont.)



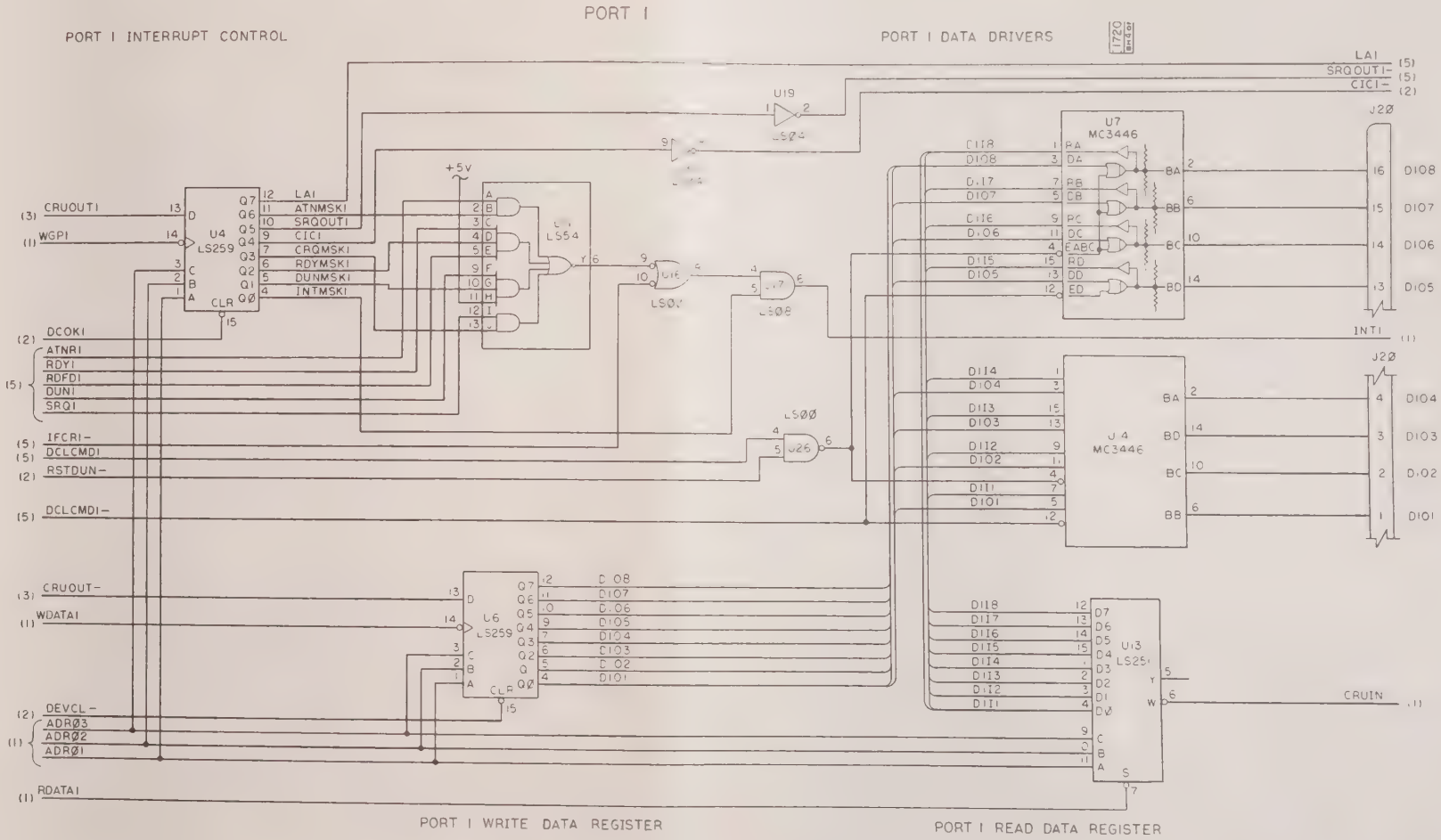
1720A-1013
(Sheet 2 of 6)

Figure 6-6. IEEE-488 Interface Module Assembly (cont.)



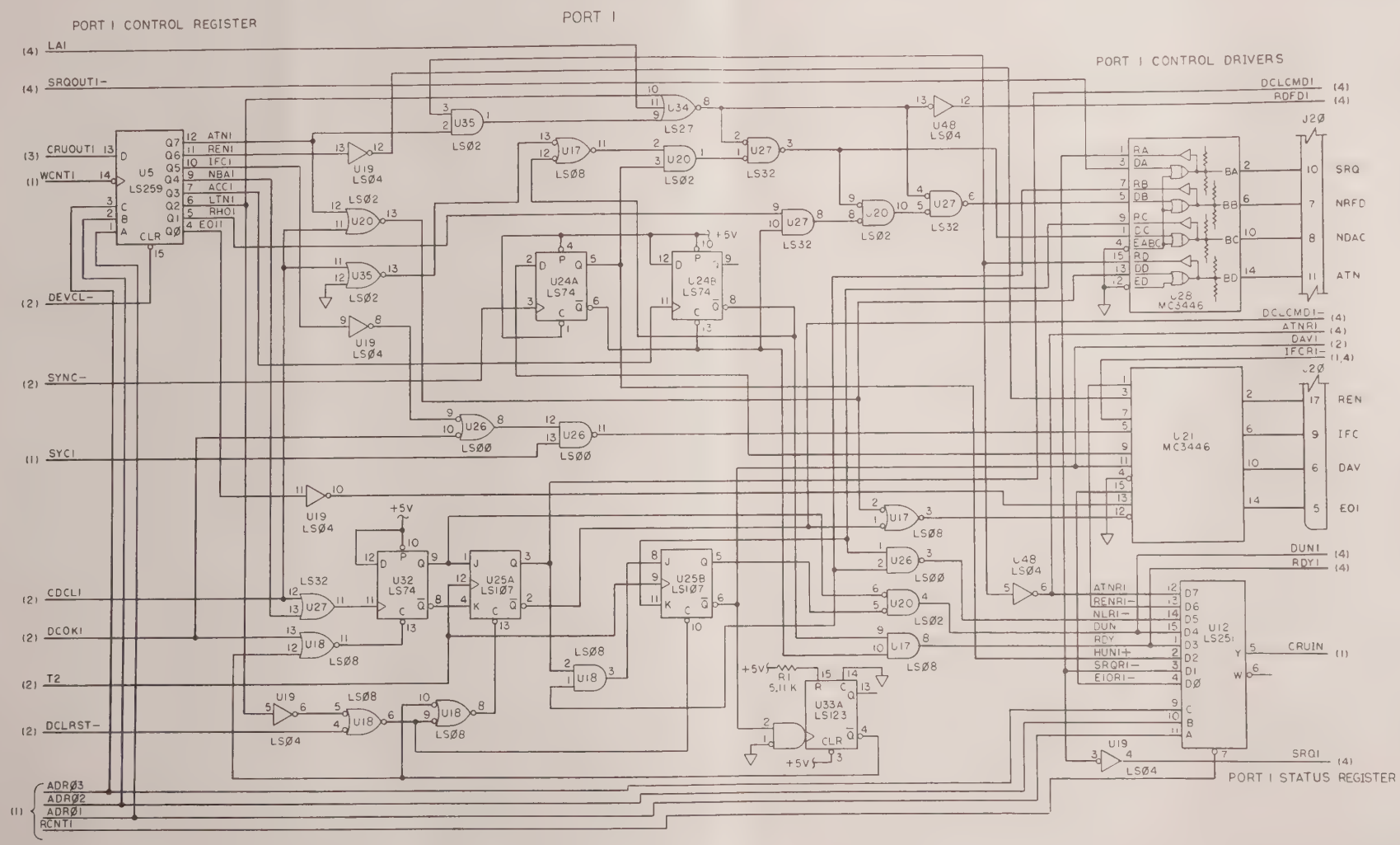
1720A-1013
(Sheet 3 of 6)

Figure 6-6. IEEE-488 Interface Module Assembly (cont.)



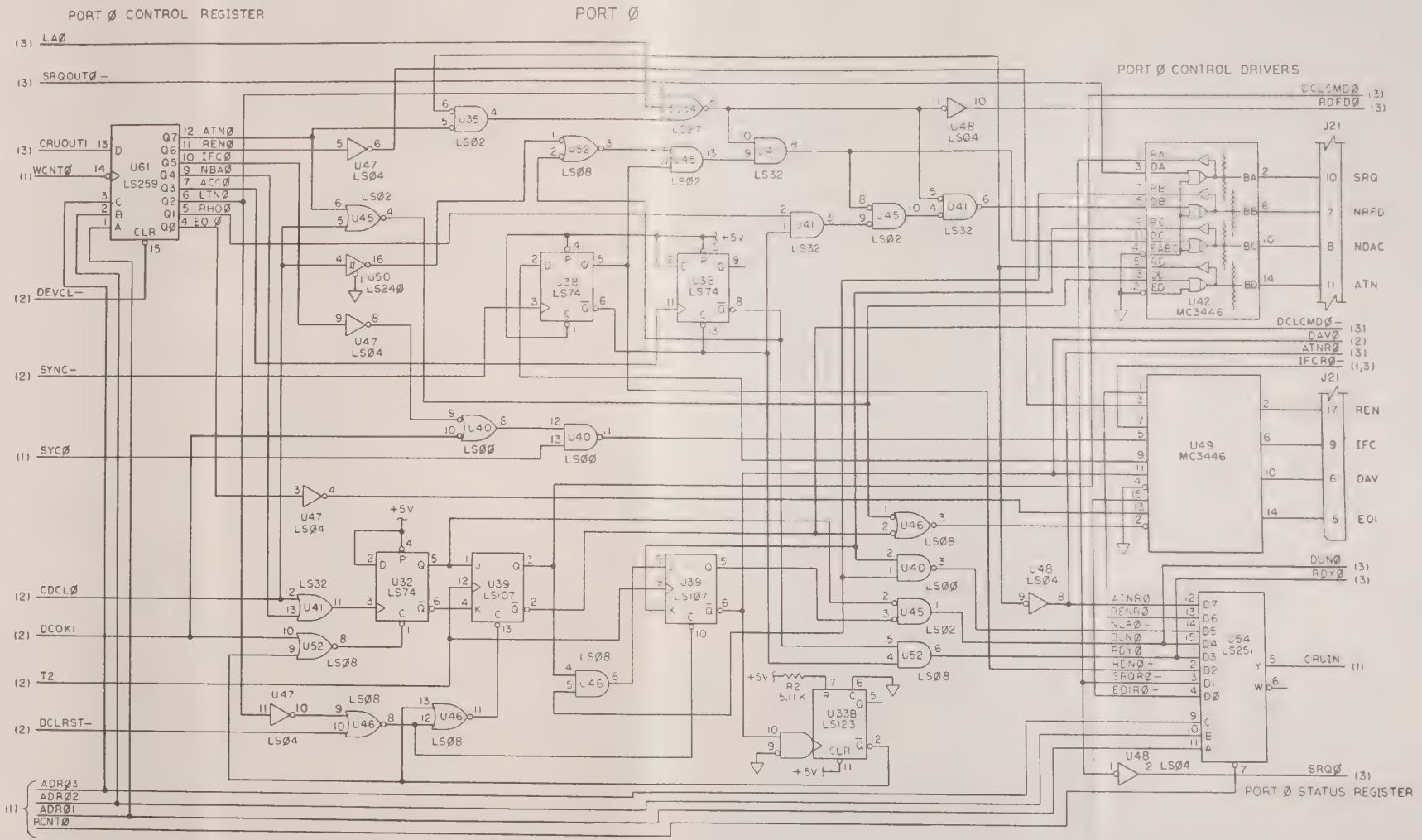
1720A-1013
(Sheet 4 of 6)

Figure 6-6. IEEE-488 Interface Module Assembly (cont.)



1720A-1013
(Sheet 5 of 6)

Figure 6-6. IEEE-488 Interface Module Assembly (cont.)



1720A-1013
(Sheet 6 of 6)

Figure 6-6. IEEE-488 Interface Module Assembly (cont.)

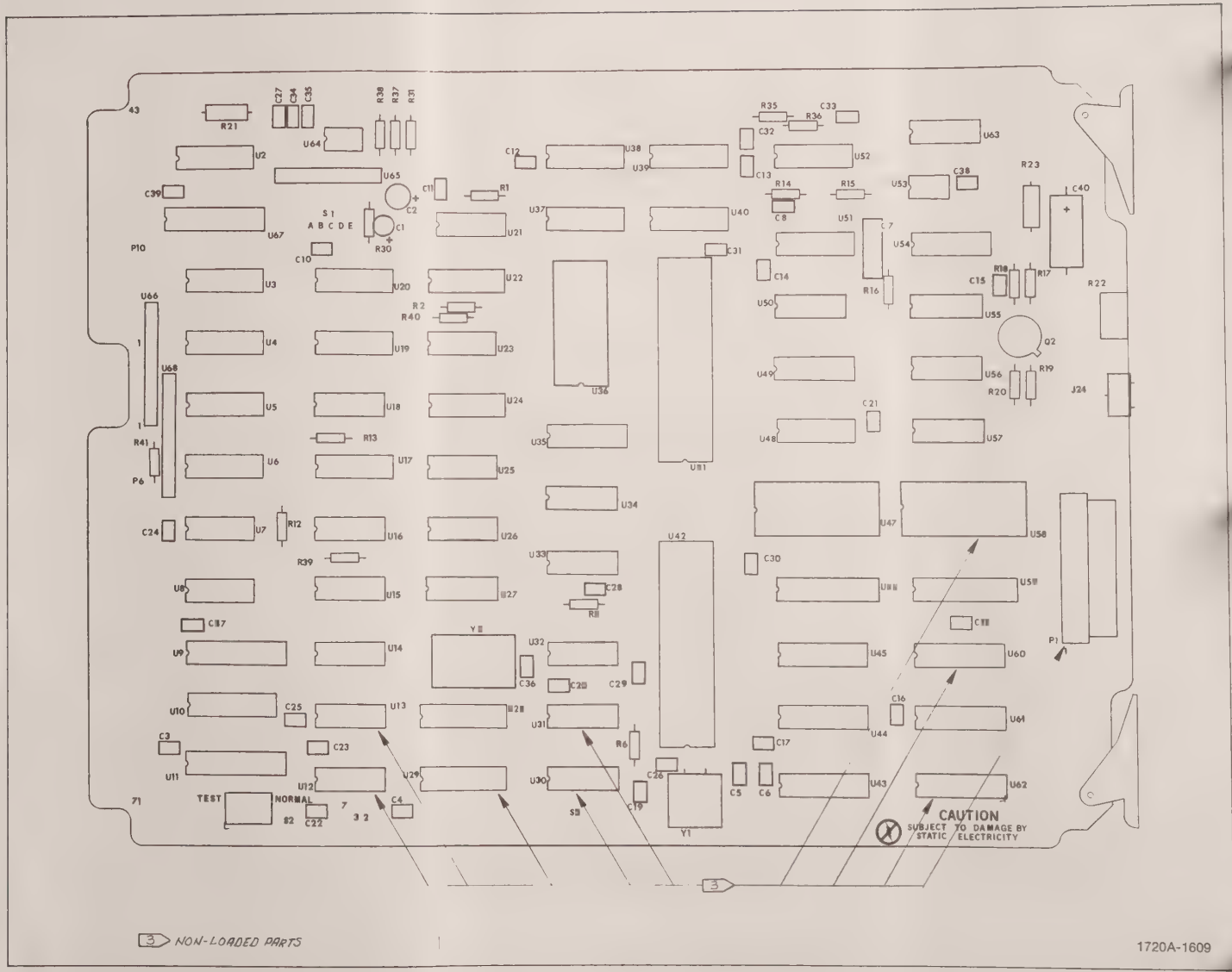
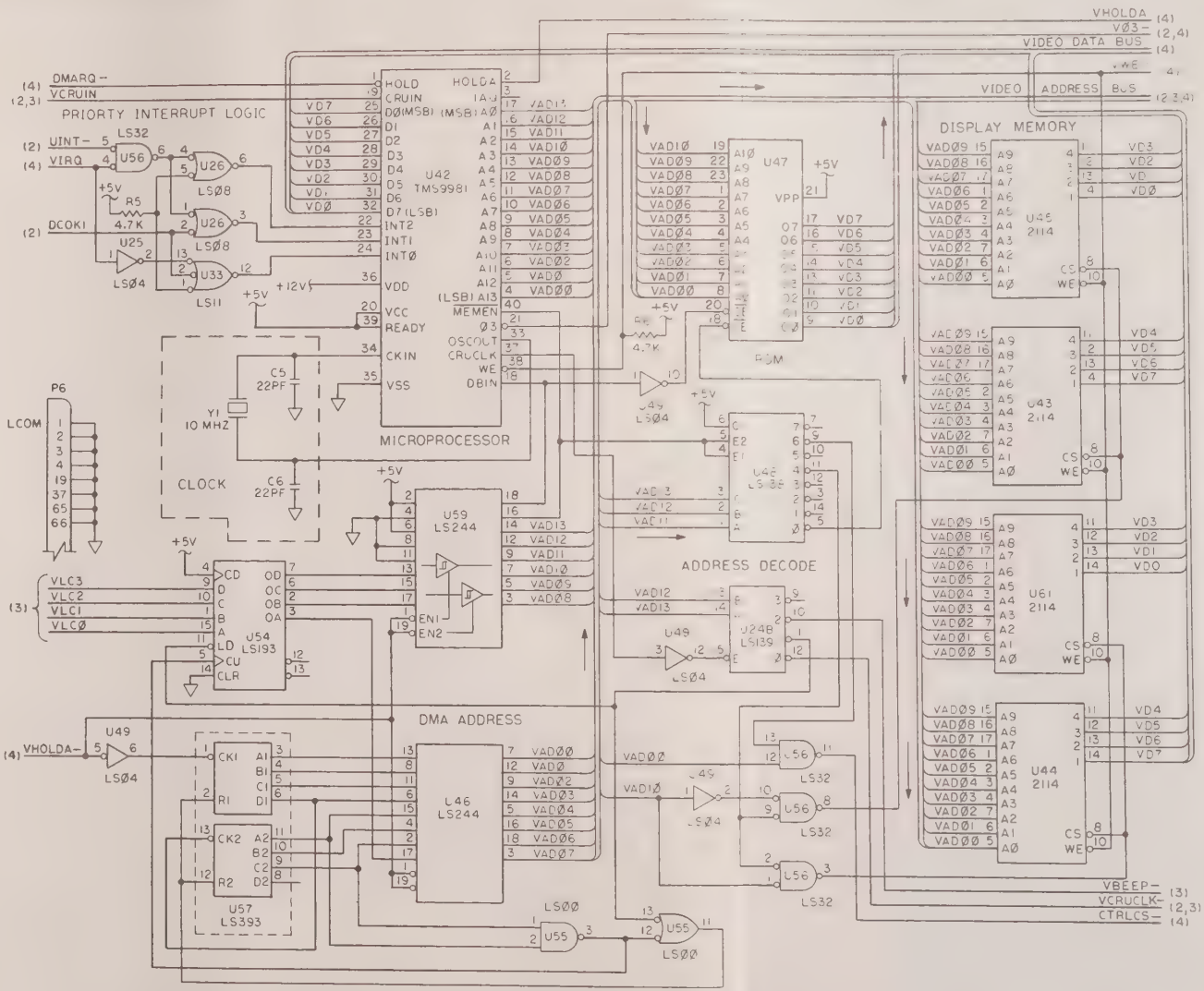


Figure 6-7. Video Keyboard Module Assembly



NOTES – UNLESS OTHERWISE SPECIFIED:

1. VADØØ LABELED AS AI3 ON 9981 (MPU-U42) IS ALSO CRUOUT ON THE CRU BUS. SEE U28-8, U2-13, U4-13, U17-13.

1720A-1009
(Sheet 1 of 4)

Figure 6-7. Video Keyboard Module Assembly (cont)

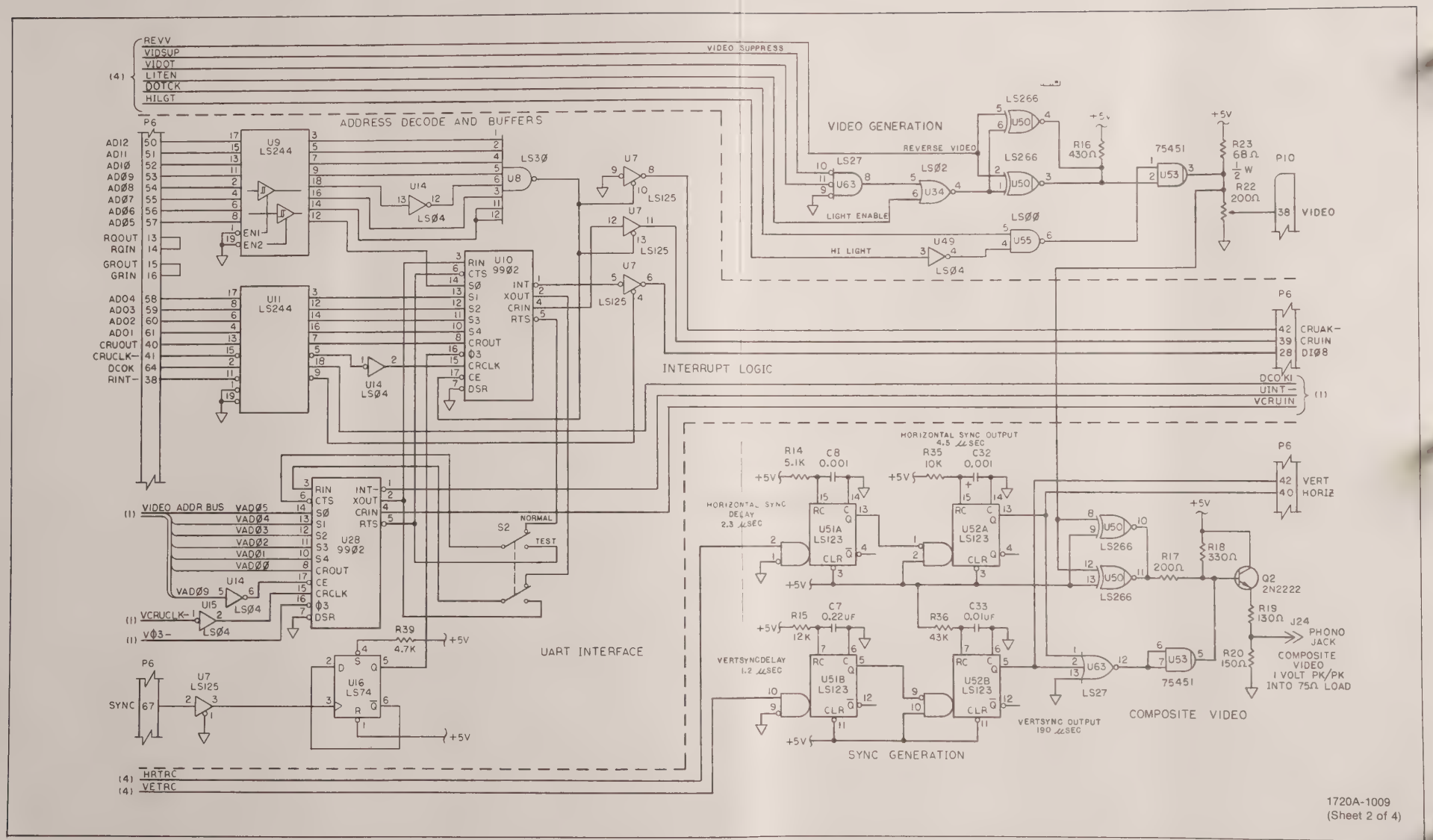
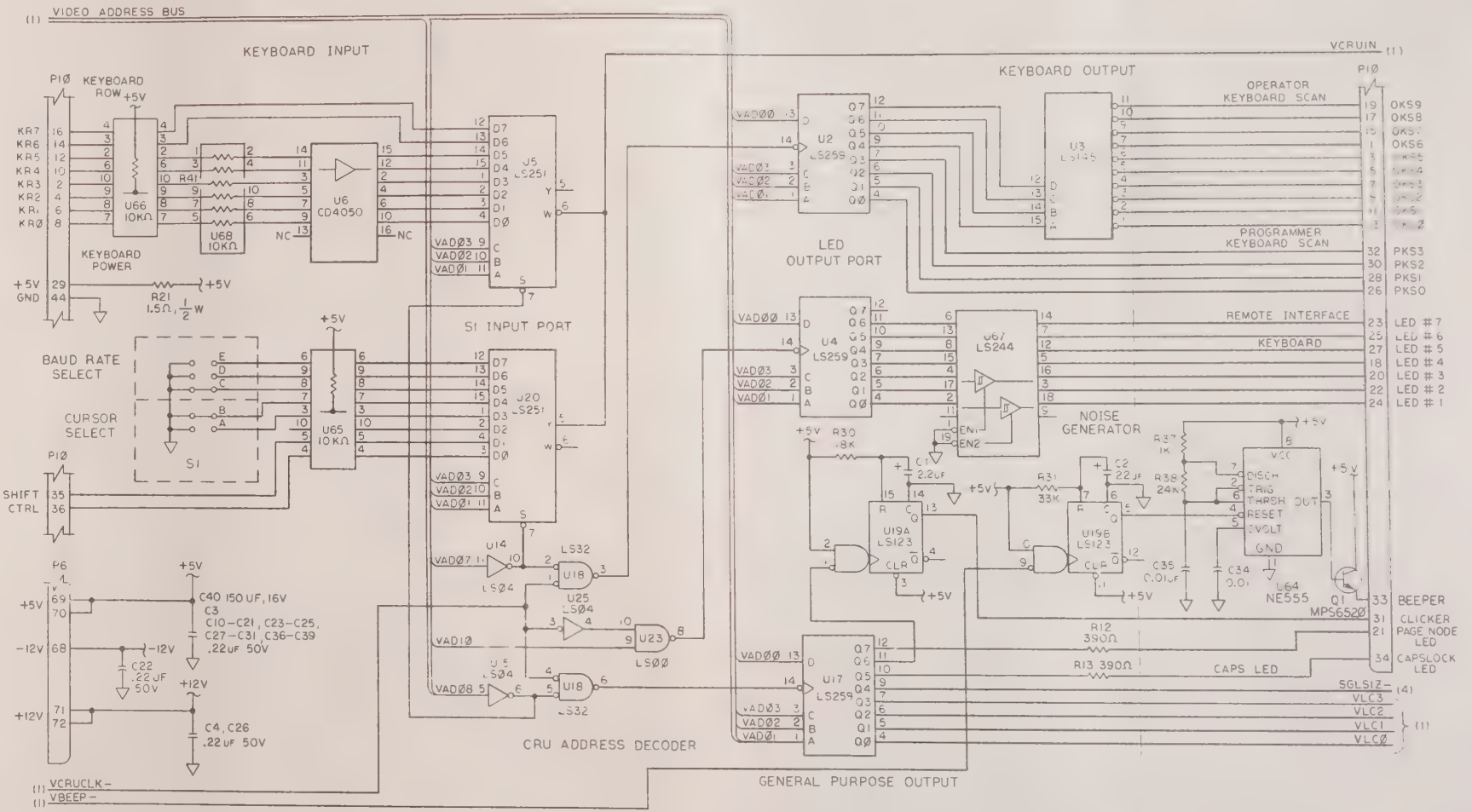


Figure 6-7. Video Keyboard Module Assembly (cont.)



1720A-1009
(Sheet 3 of 4)

Figure 6-7. Video Keyboard Module Assembly (cont.)

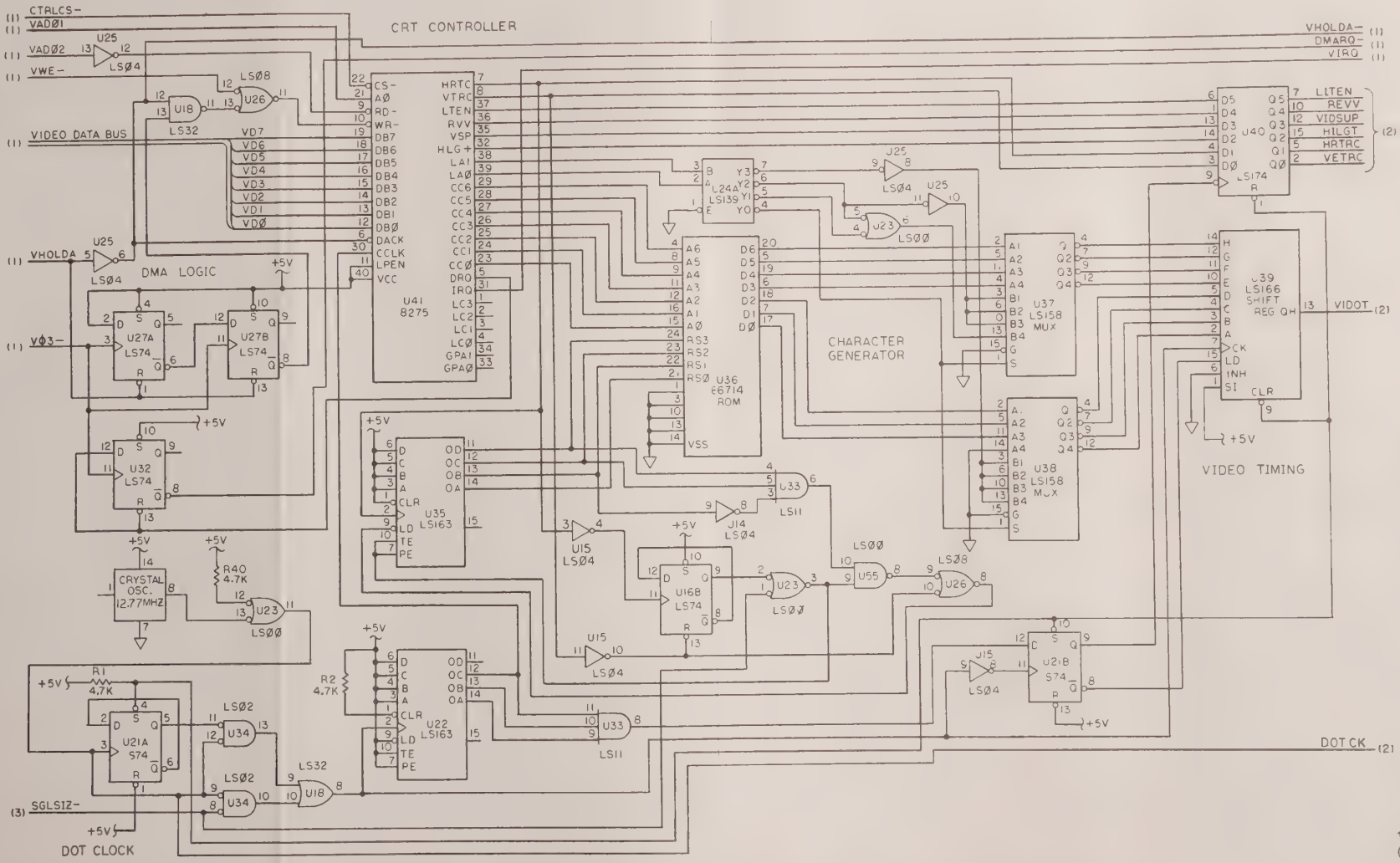


Figure 6-7. Video Keyboard Module Assembly (cont.)

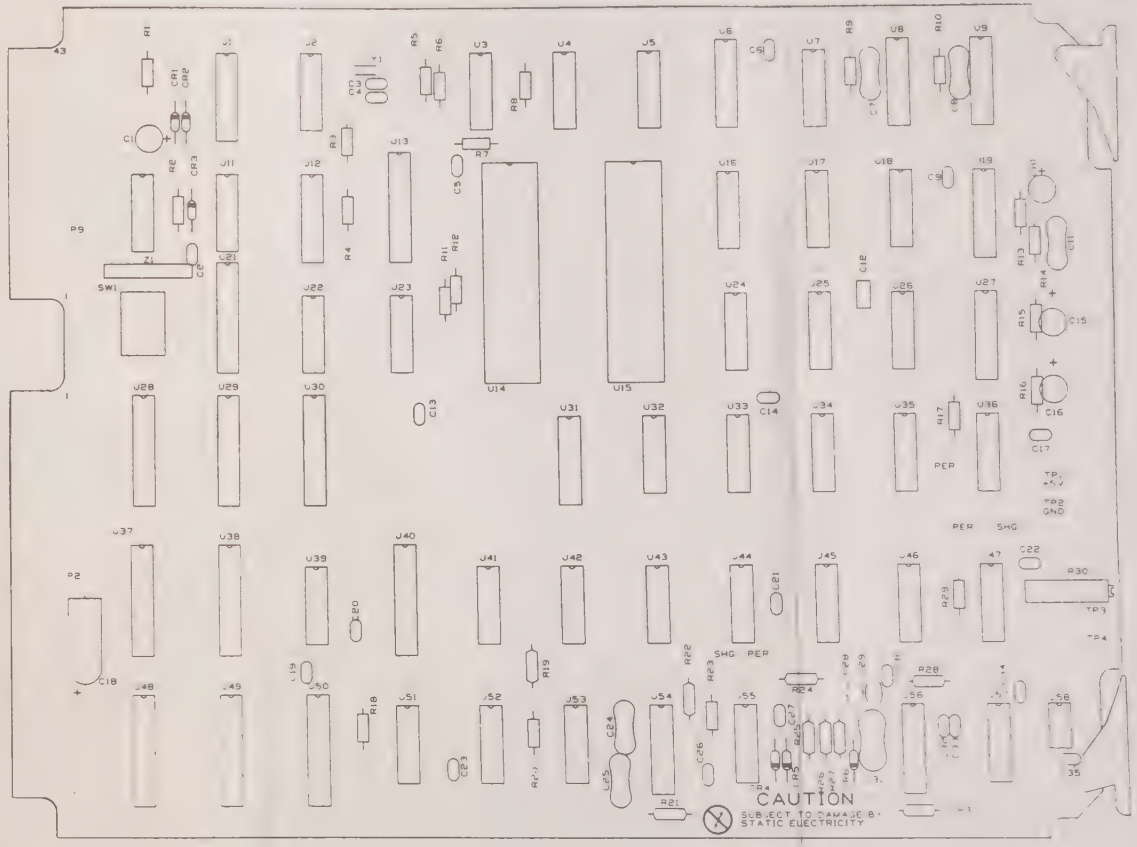


Figure 6-8. DMA Floppy/Clock Module Assembly

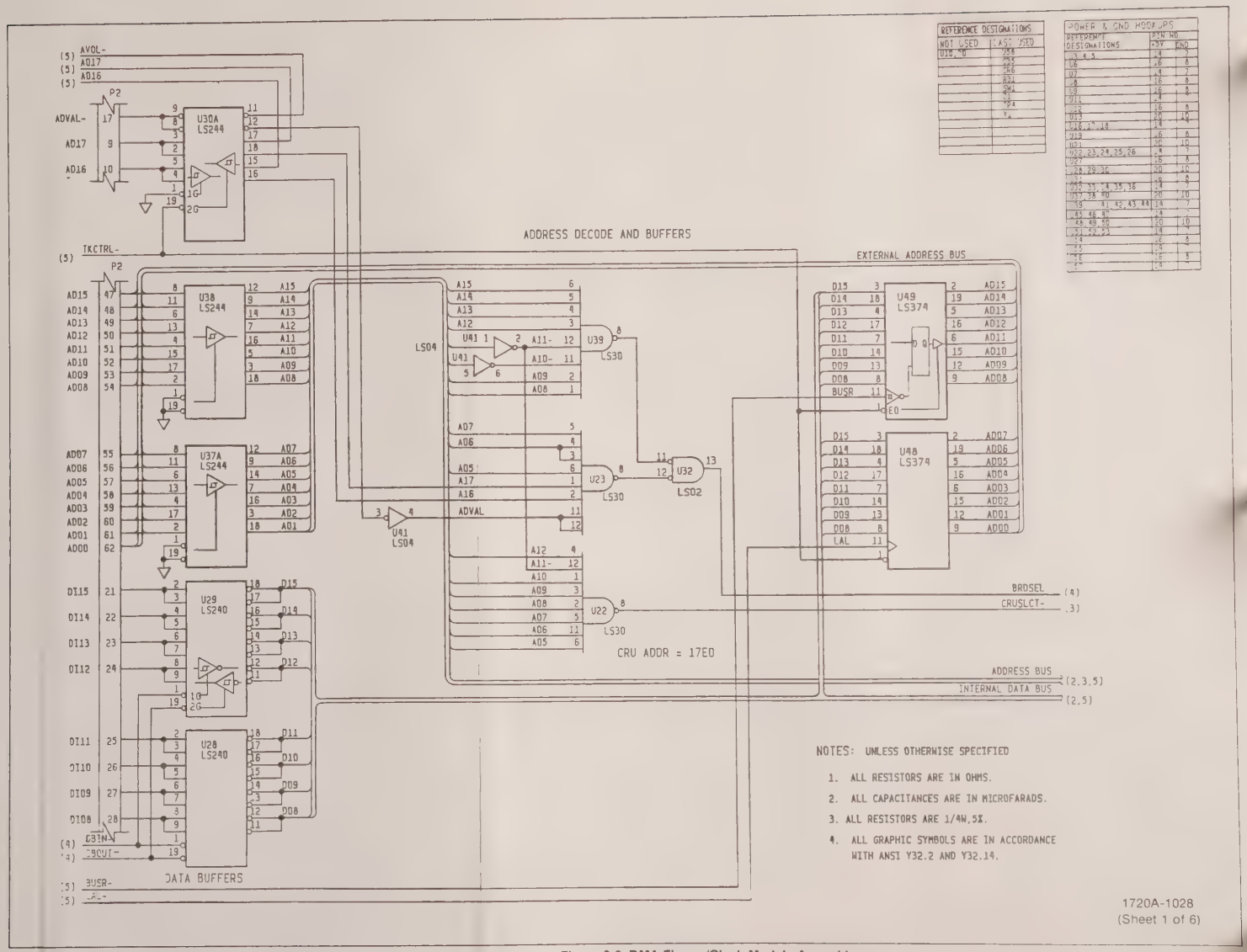
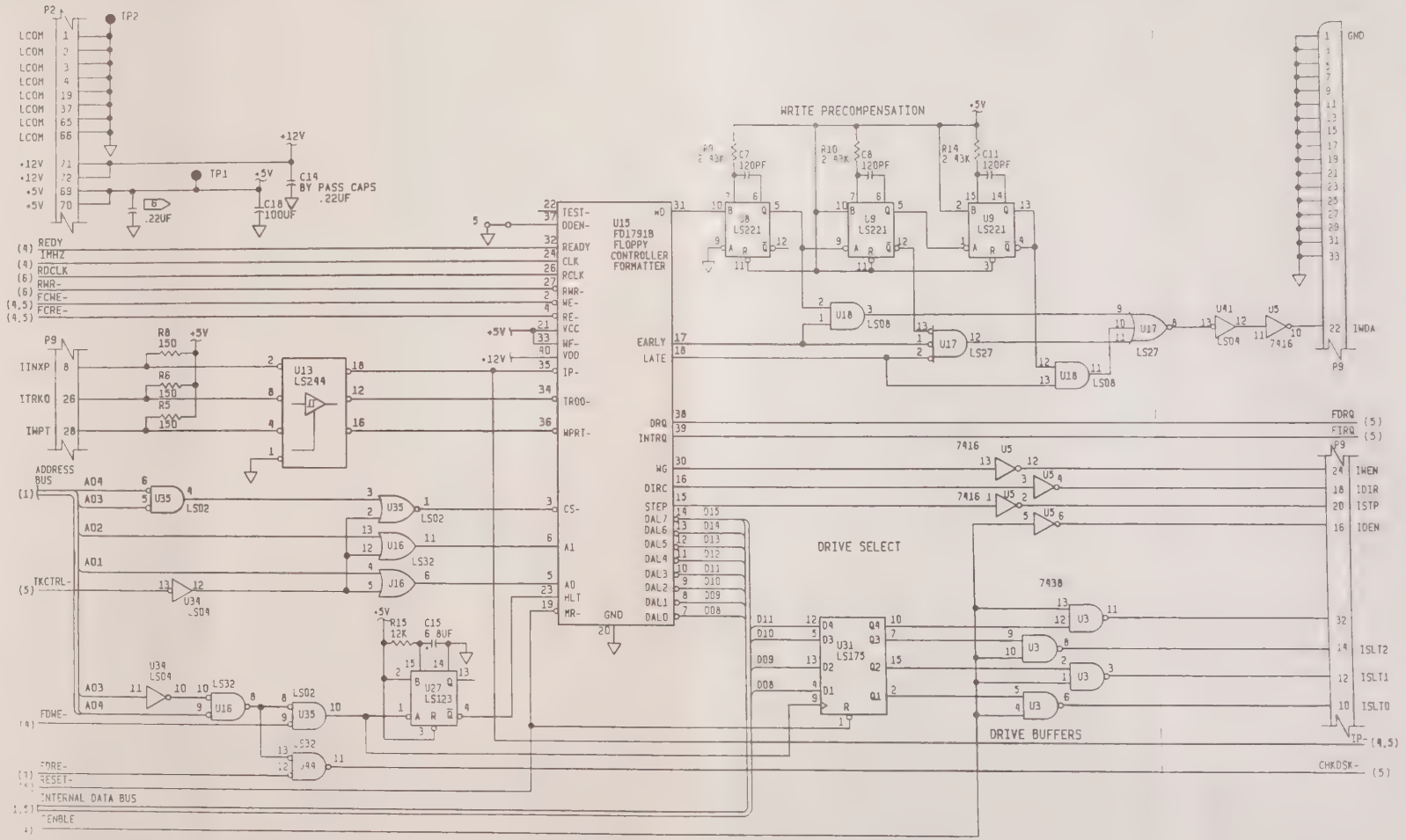
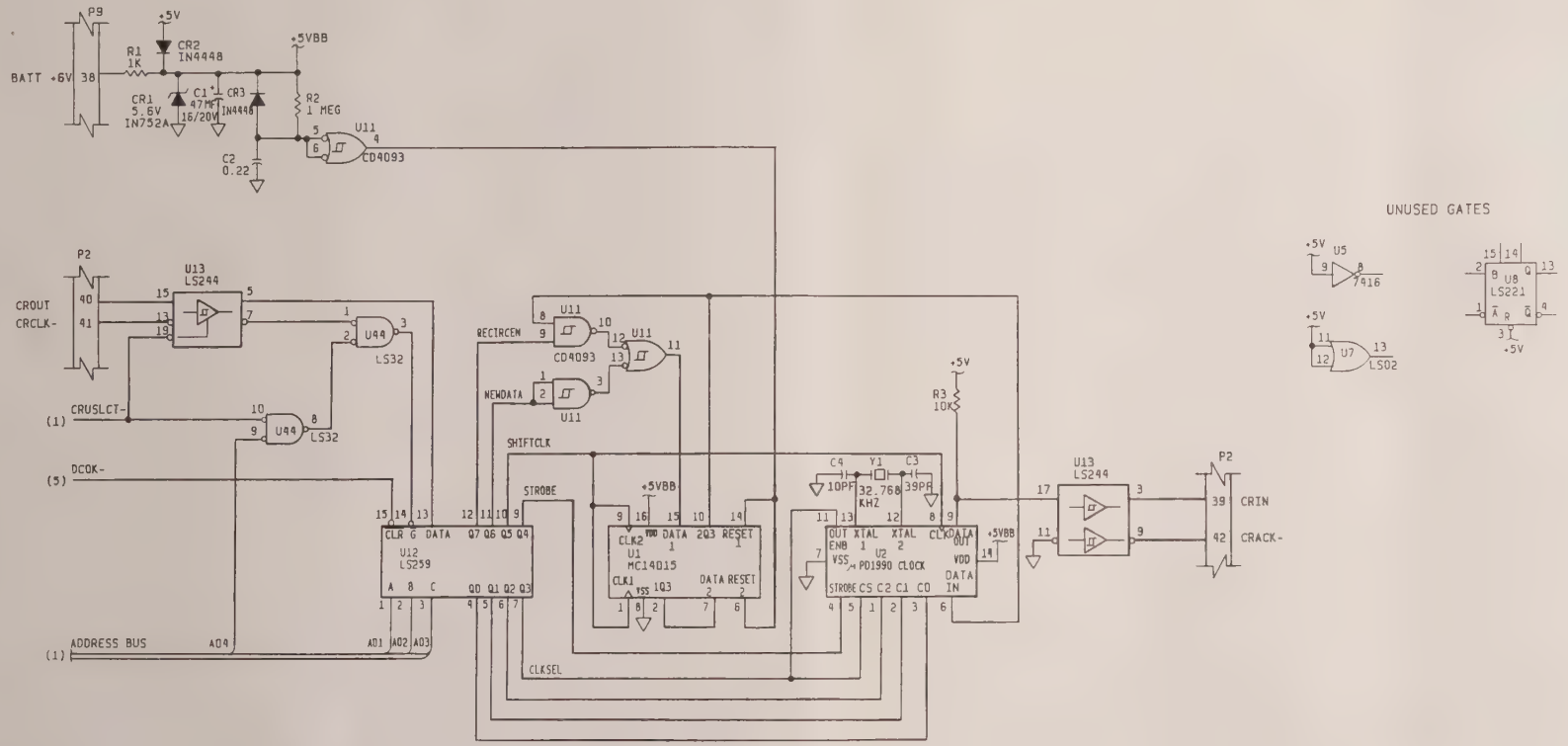


Figure 6-8. DMA Floppy/Clock Module Assembly (cont.)



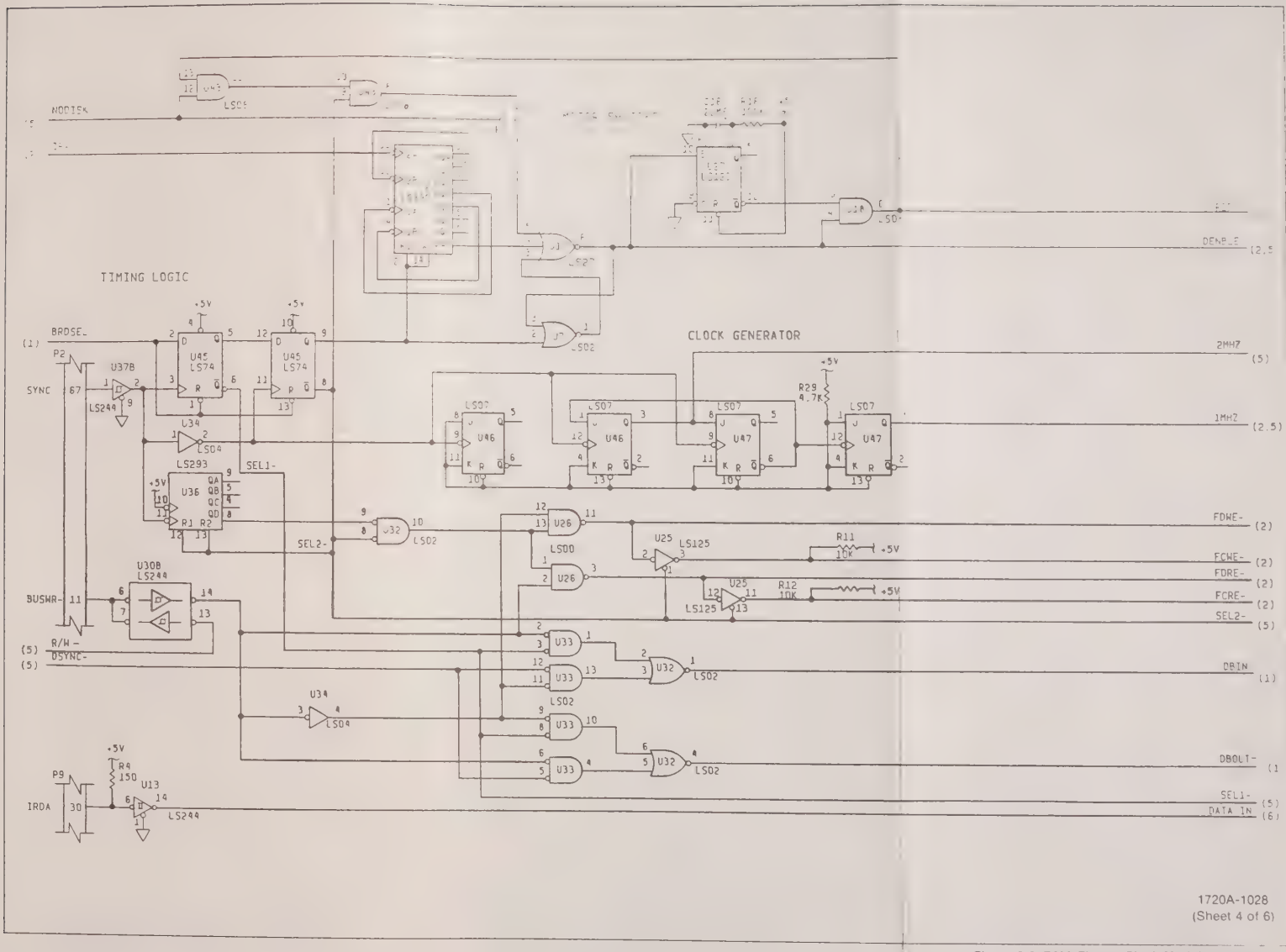
1720A-1028
(Sheet 2 of 6)

Figure 6-8. DMA Floppy/Clock Module Assembly (cont.)



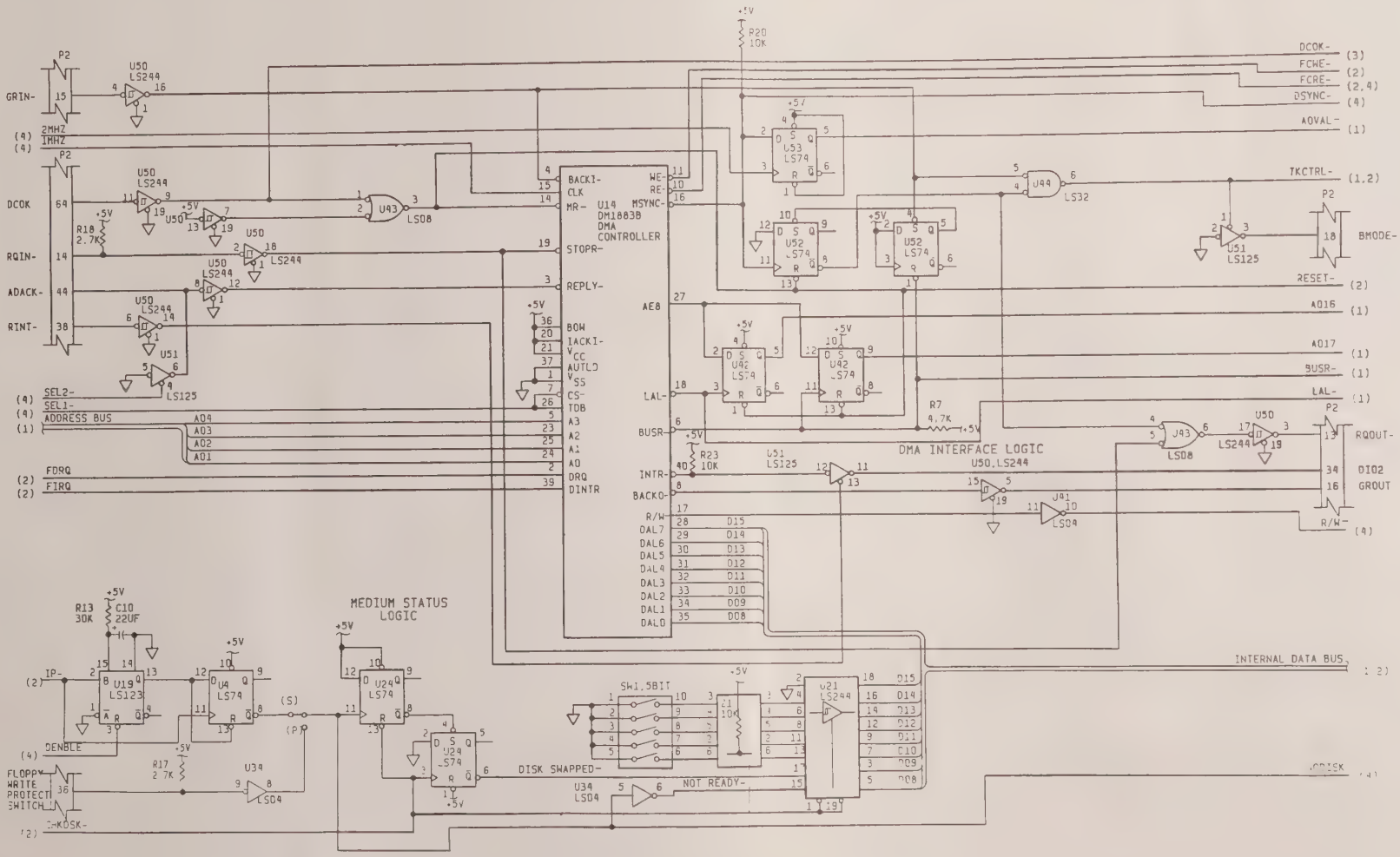
1720A-1028
(Sheet 3 of 6)

Figure 6-8. DMA Floppy/Clock Module Assembly (cont.)



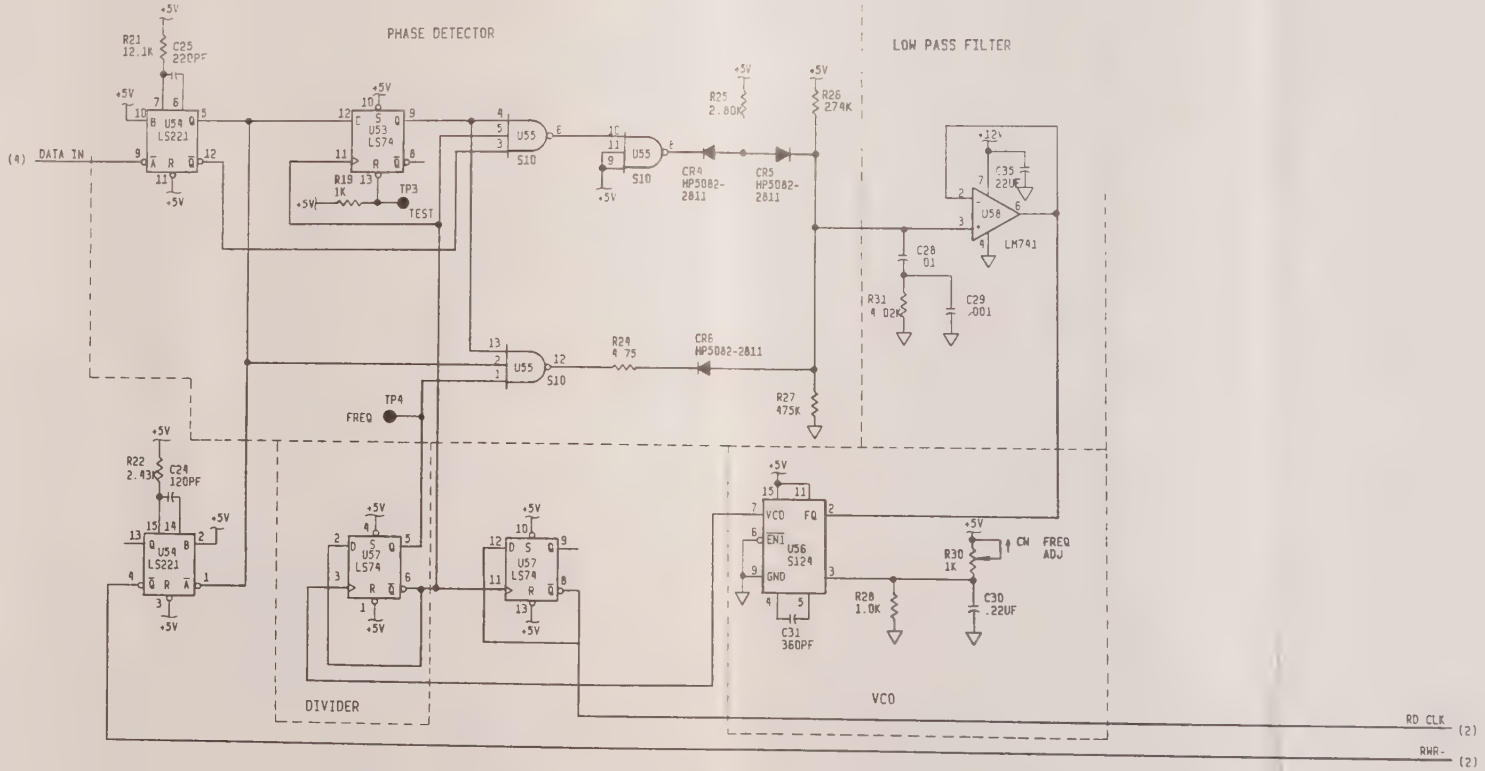
1720A-1028
(Sheet 4 of 6)

Figure 6-8. DMA Floppy/Clock Module Assembly (cont.)



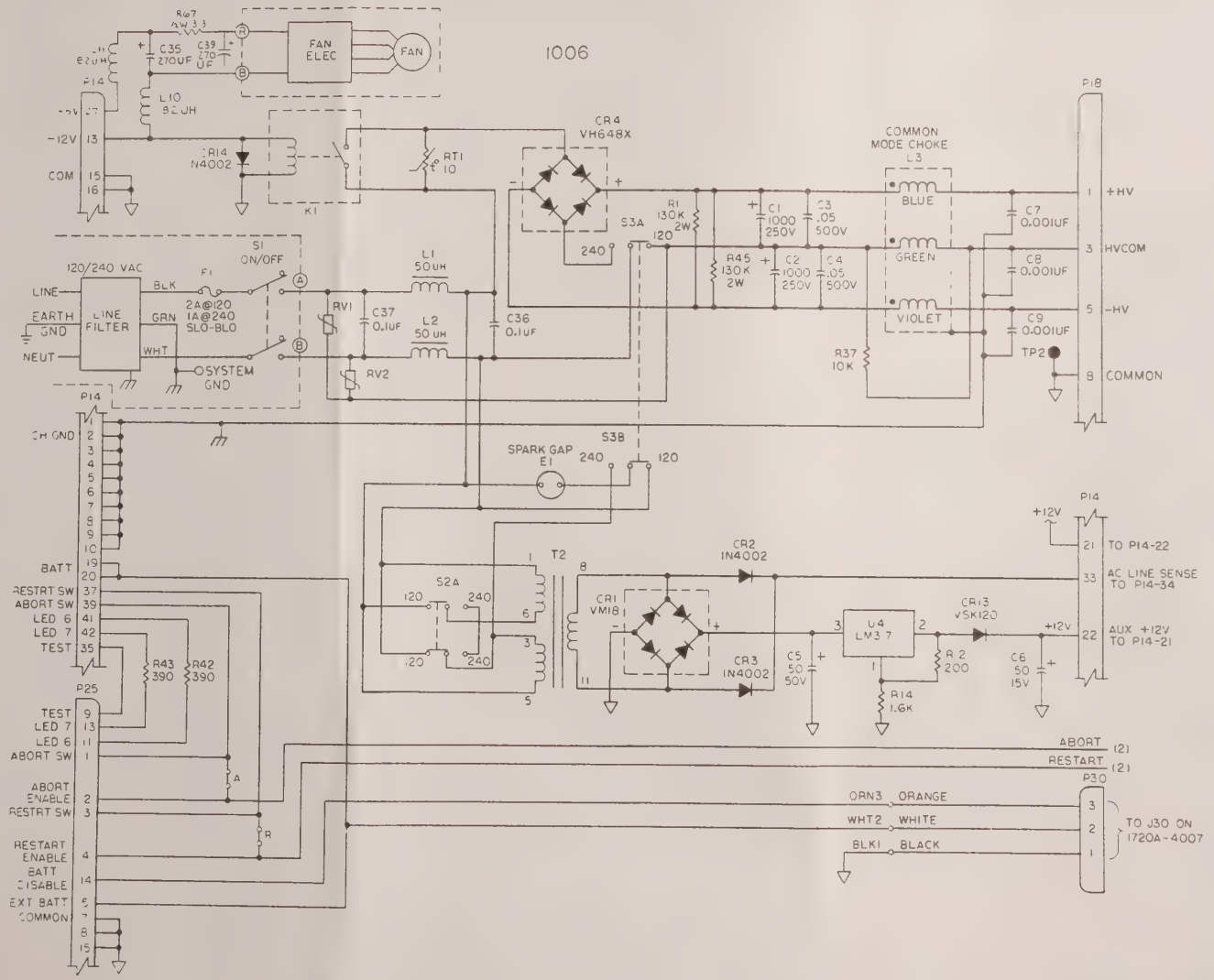
1720A-1028
(Sheet 5 of 6)

Figure 6-8. DMA Floppy/Clock Module Assembly (cont.)



1720A-1028
(Sheet 6 of 6)

Figure 6-8. DMA Floppy/Clock Module Assembly (cont.)

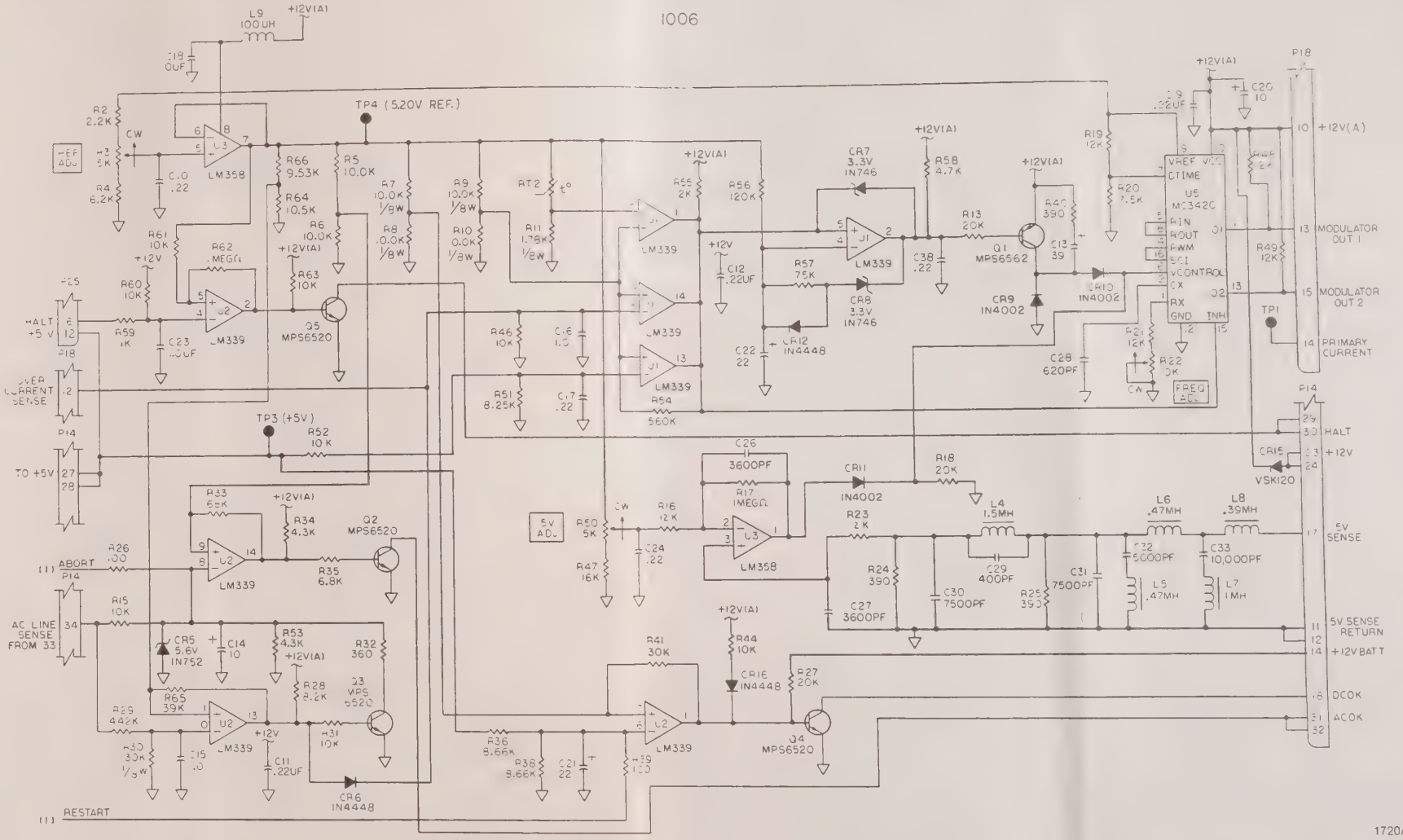


NOTES—UNLESS OTHERWISE SPECIFIED:

1. THIS SCHEMATIC CONTAINS ALL THE INFORMATION THAT WOULD NORMALLY APPEAR ON THE 1720A 1006, 1720A-1007, 1720A-1016.
2. ALL RESISTANCES ARE IN OHMS.
3. ALL CAPACITANCES ARE IN MICROFARADS.
4. (A) AFTER VOLTAGE INDICATES AUXILIARY SUPPLY.

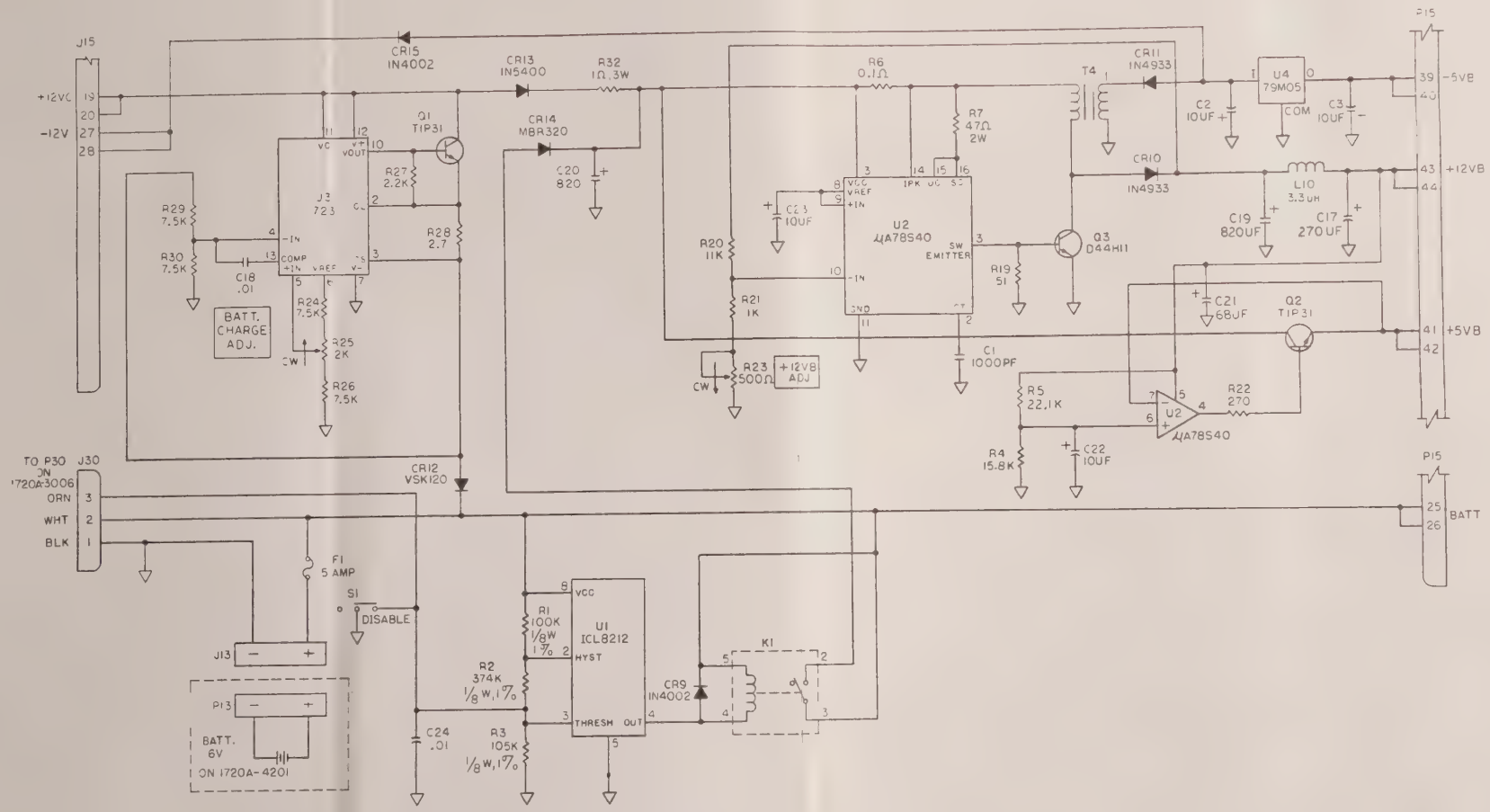
1720A-1201
(Sheet 1 of 4)

Figure 6-9. Power Supply Module Assembly



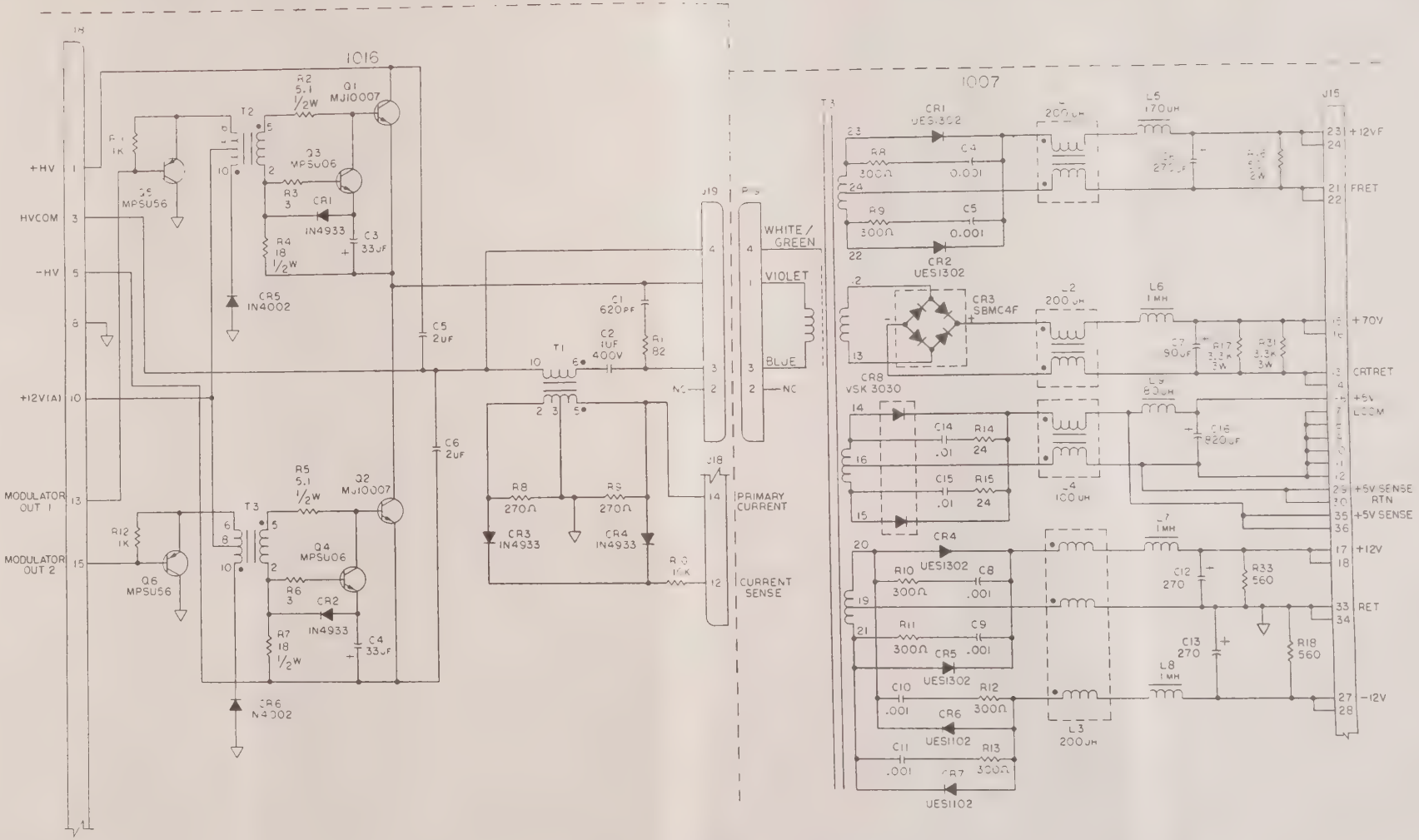
1720A-1201
(Sheet 2 of 4)

Figure 6-9. Power Supply Module Assembly (cont.)



1720A-1201
(Sheet 3 of 4)

Figure 6-9. Power Supply Module Assembly (cont.)



1720A-1201
(Sheet 4 of 4)

Figure 6-9. Power Supply Module Assembly (cont.)

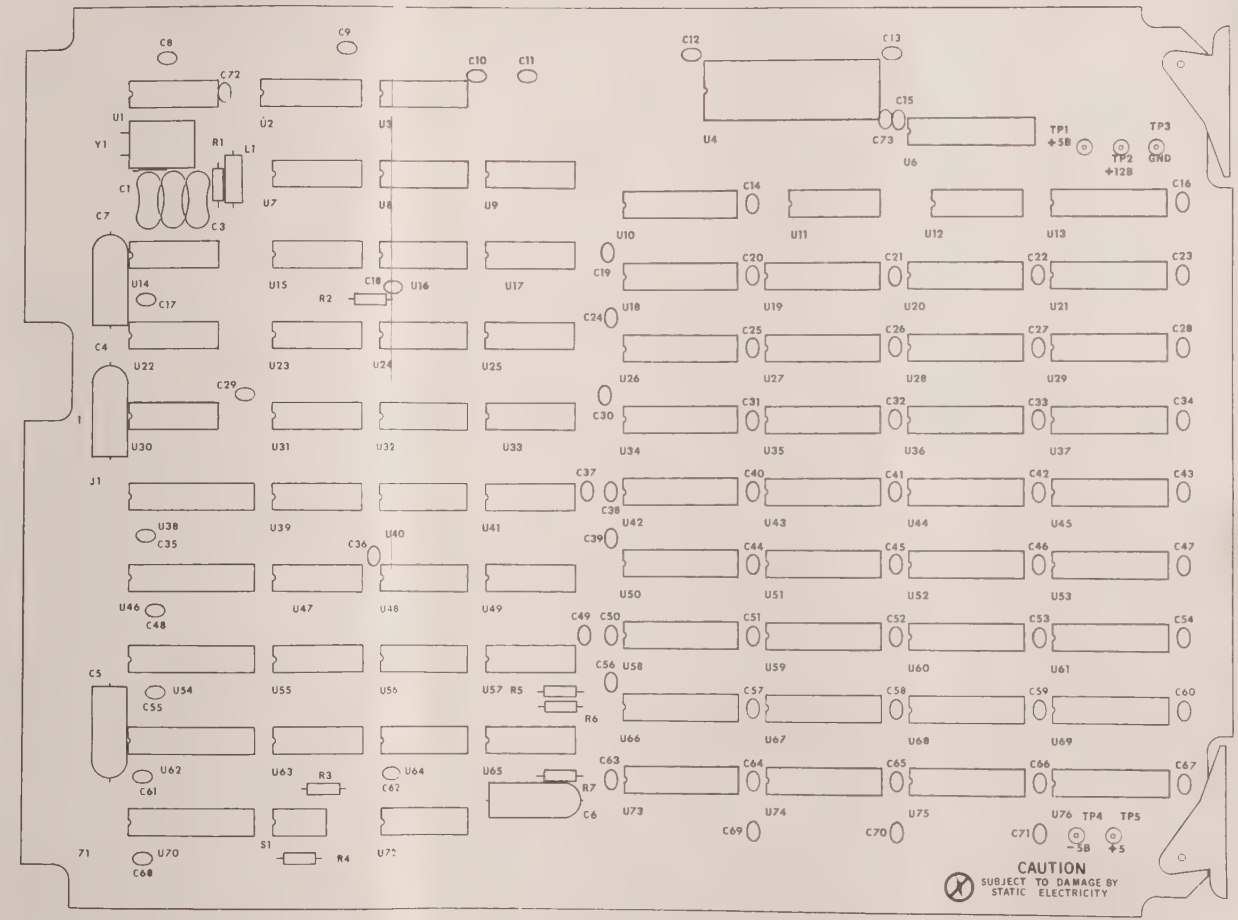


Figure 6-10. Electronic Disk Module Assembly

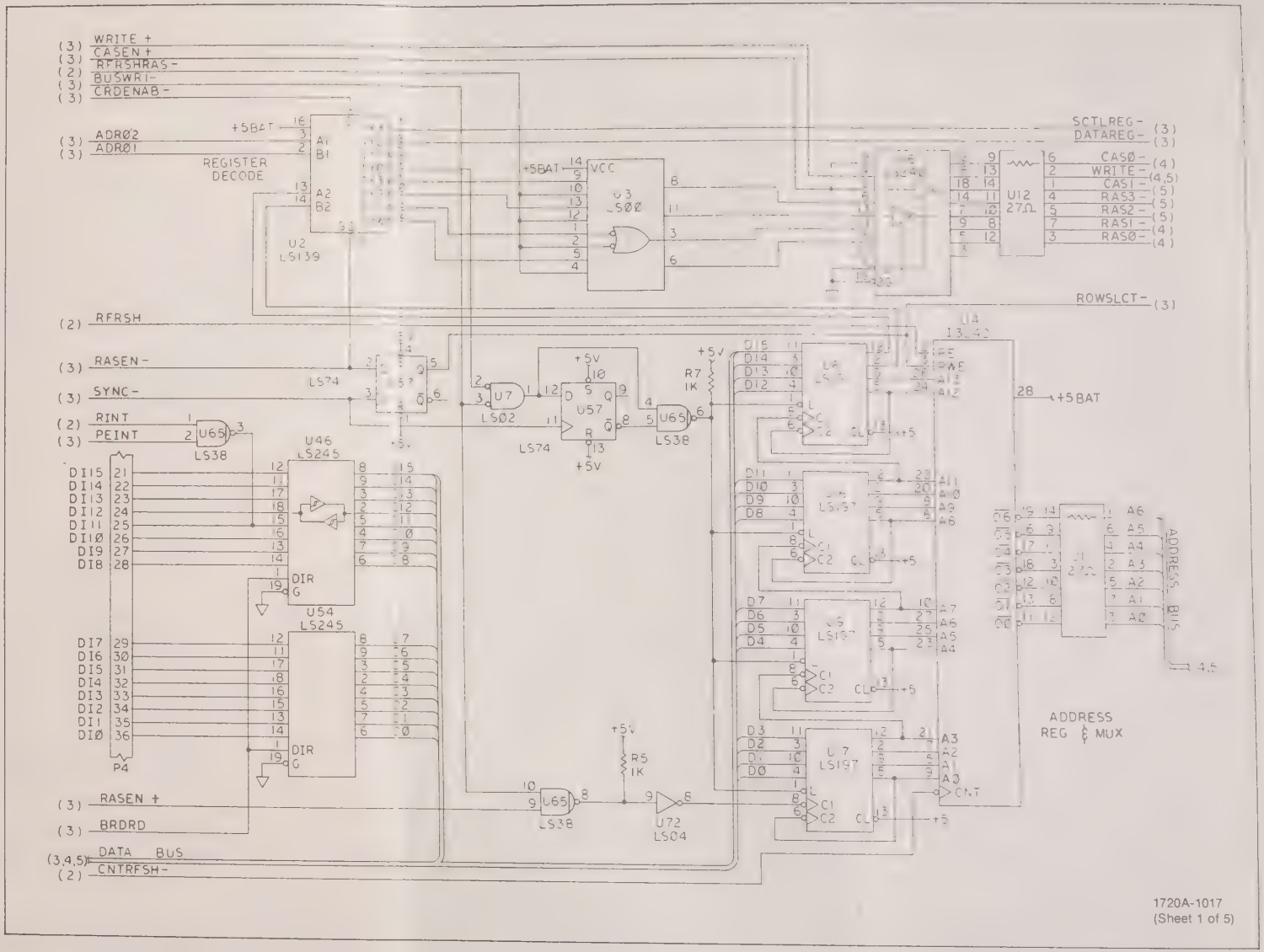
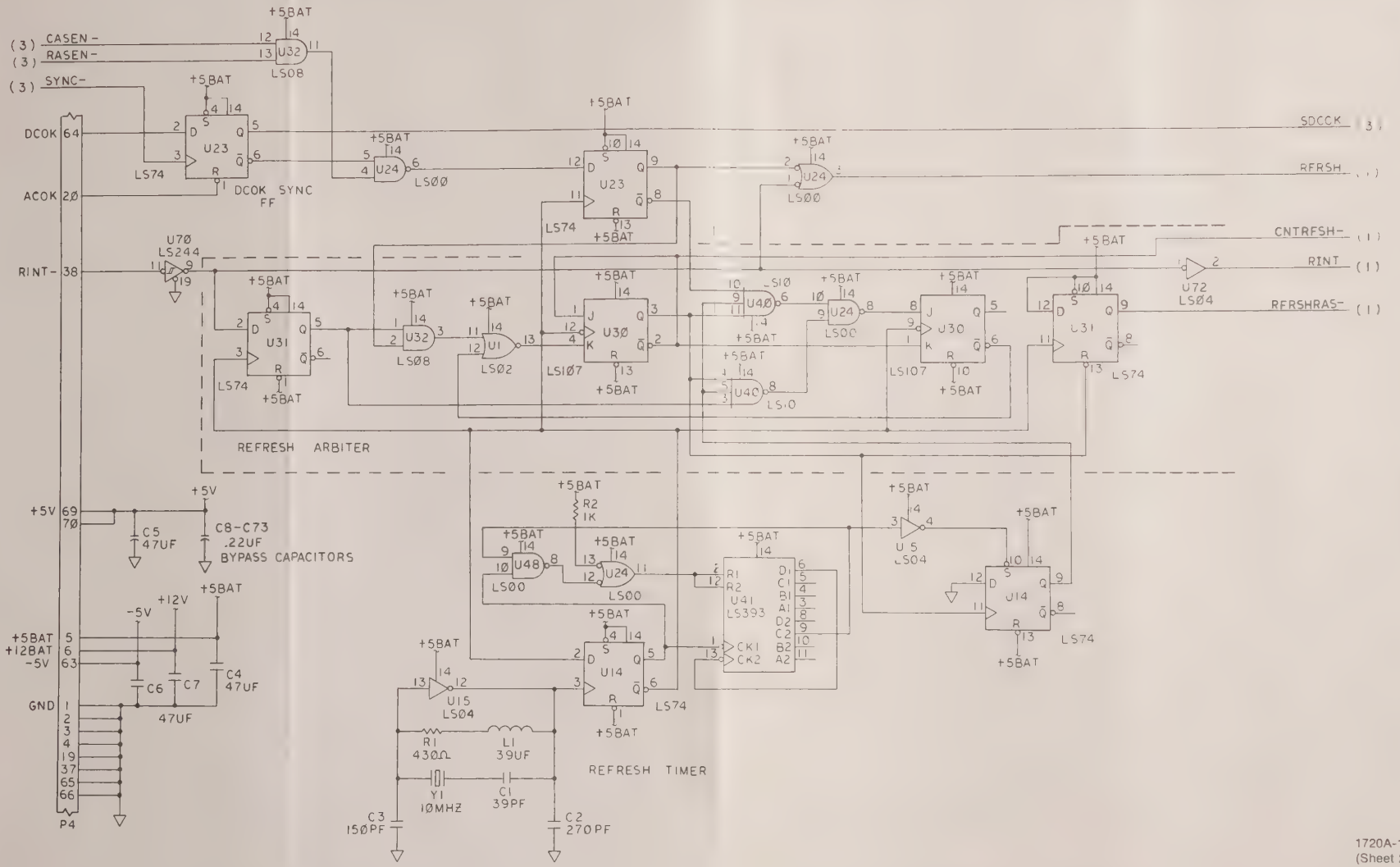
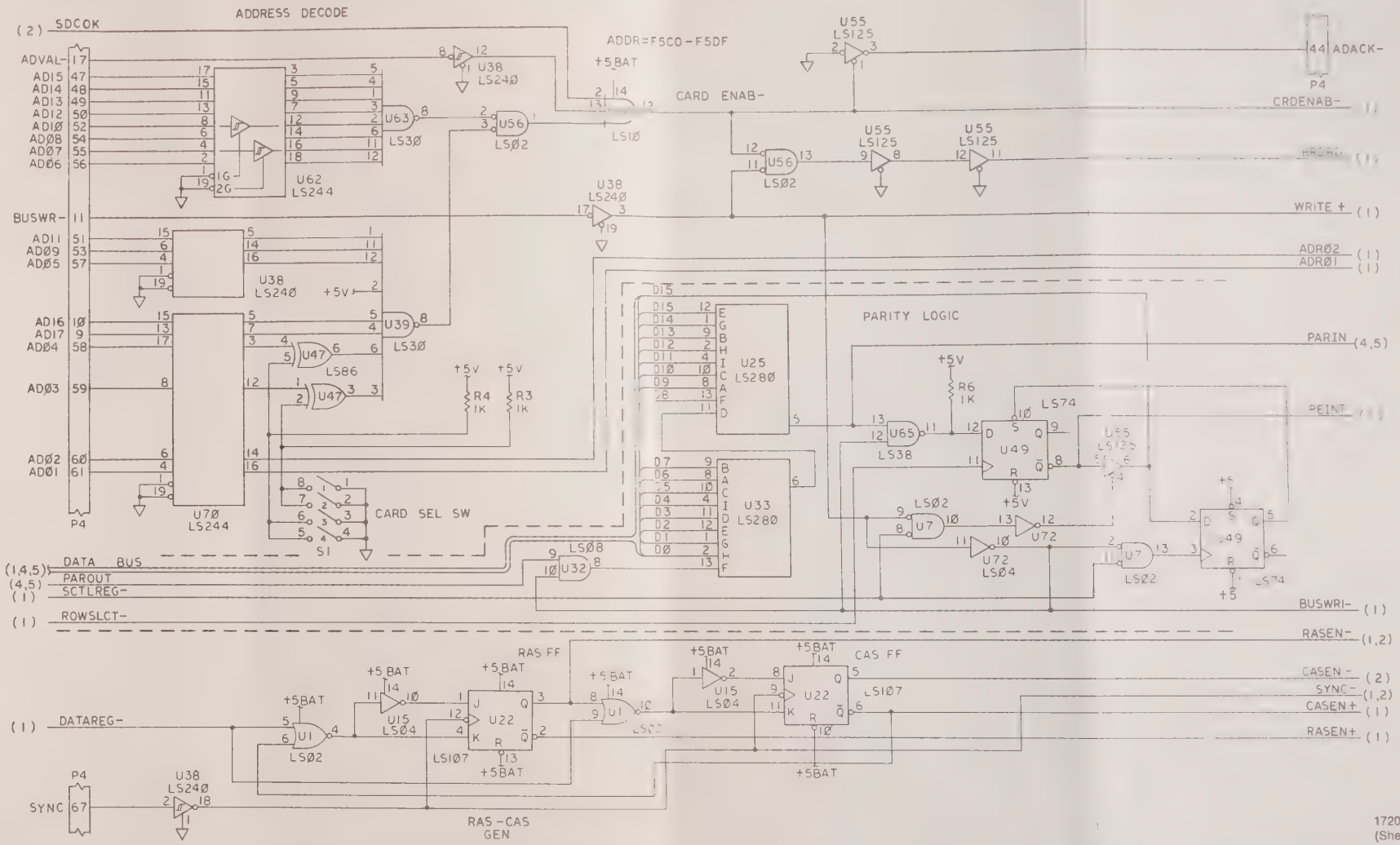


Figure 6-10. Electronic Disk Module Assembly (cont.)



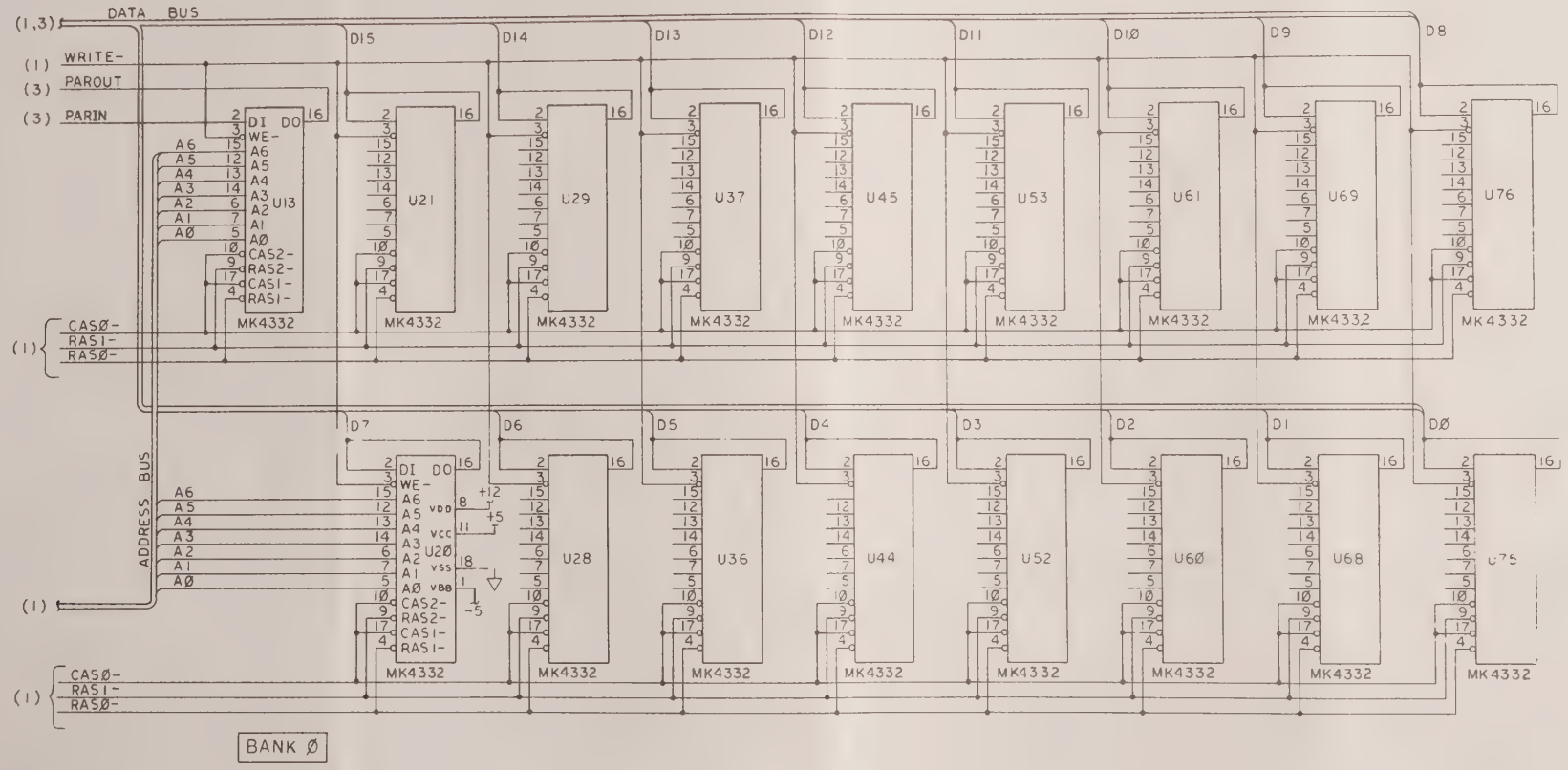
1720A-1017
(Sheet 2 of 5)

Figure 6-10. Electronic Disk Module Assembly (cont.)



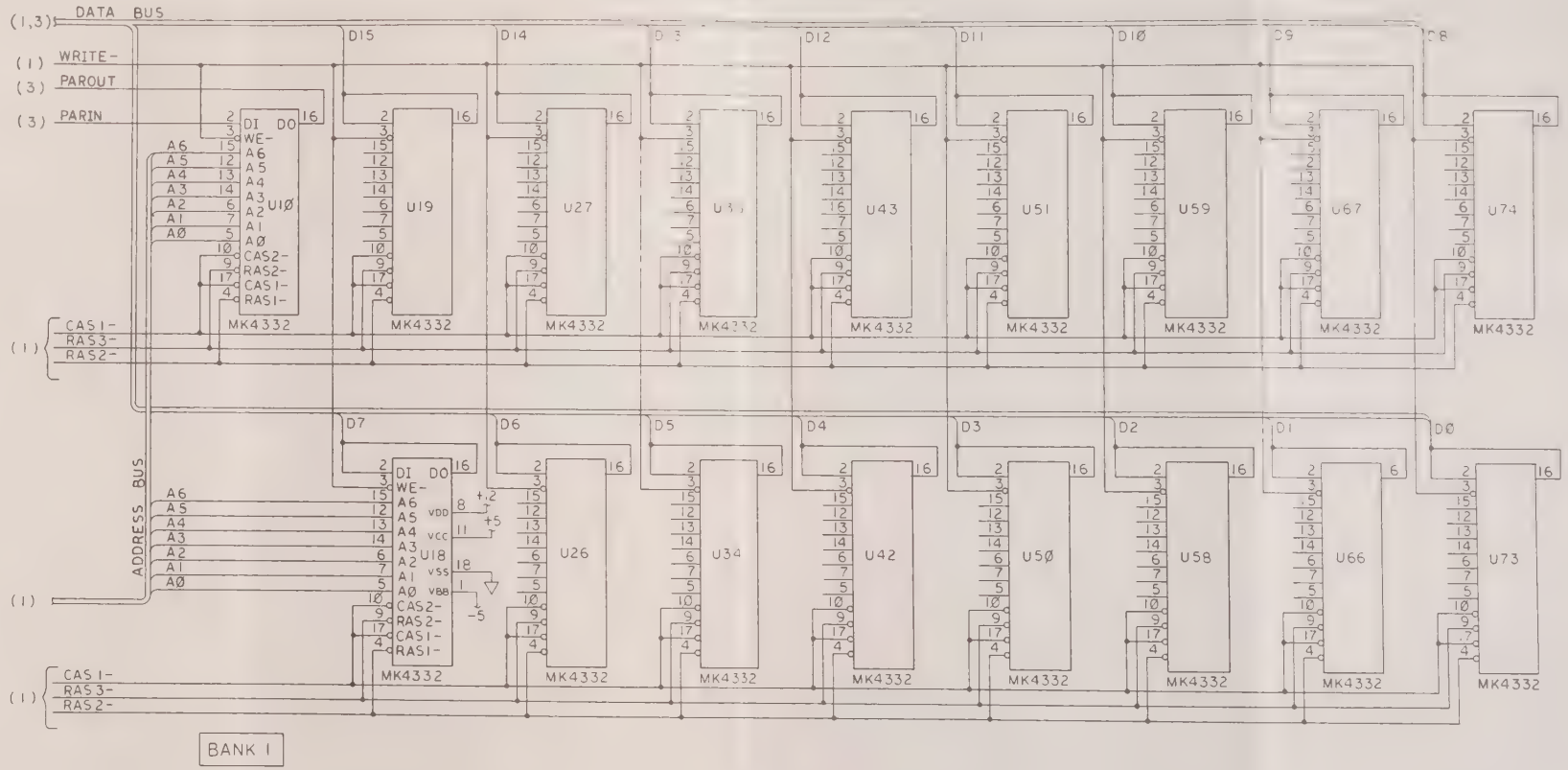
1720A-1017
(Sheet 3 of 5)

Figure 6-10. Electronic Disk Module Assembly (cont.)



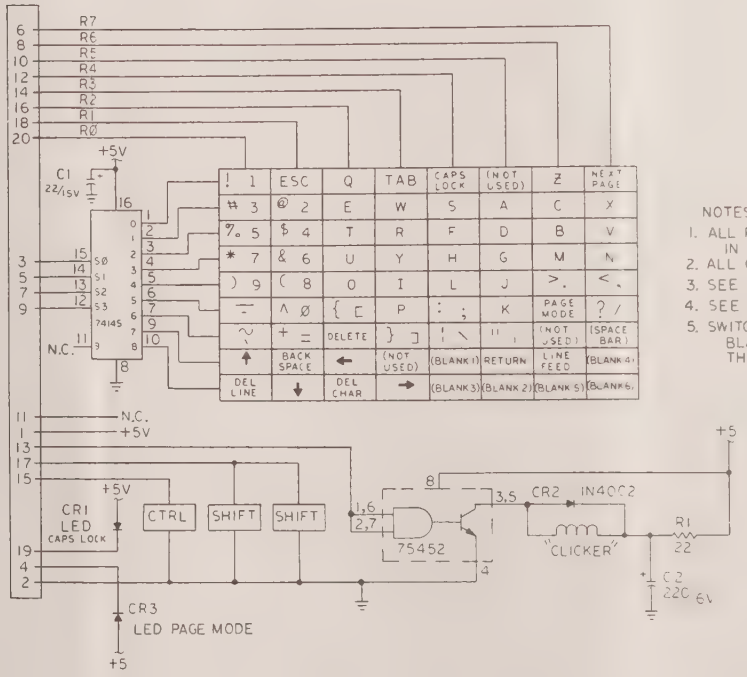
1720A-1017
(Sheet 4 of 5)

Figure 6-10. Electronic Disk Module Assembly (cont.)



1720A-1017
(Sheet 5 of 5)

Figure 6-10. Electronic Disk Module Assembly (cont.)



NOTES-UNLESS OTHERWISE SPECIFIED:

1. ALL RESISTOR VALUES ARE GIVEN IN OHMS 1/4W.
2. ALL CAPACITOR VALUES ARE IN μF.
3. SEE 1720A-3010 FOR P.C.B.
4. SEE 1720A-8001 FOR ASSY.
5. SWITCH POSITIONS BLANK 1 THROUGH BLANK 6 ARE NOT DECODED BY THE 1720A.

1720A-1010

Figure 6-11. Keyboard Schematic

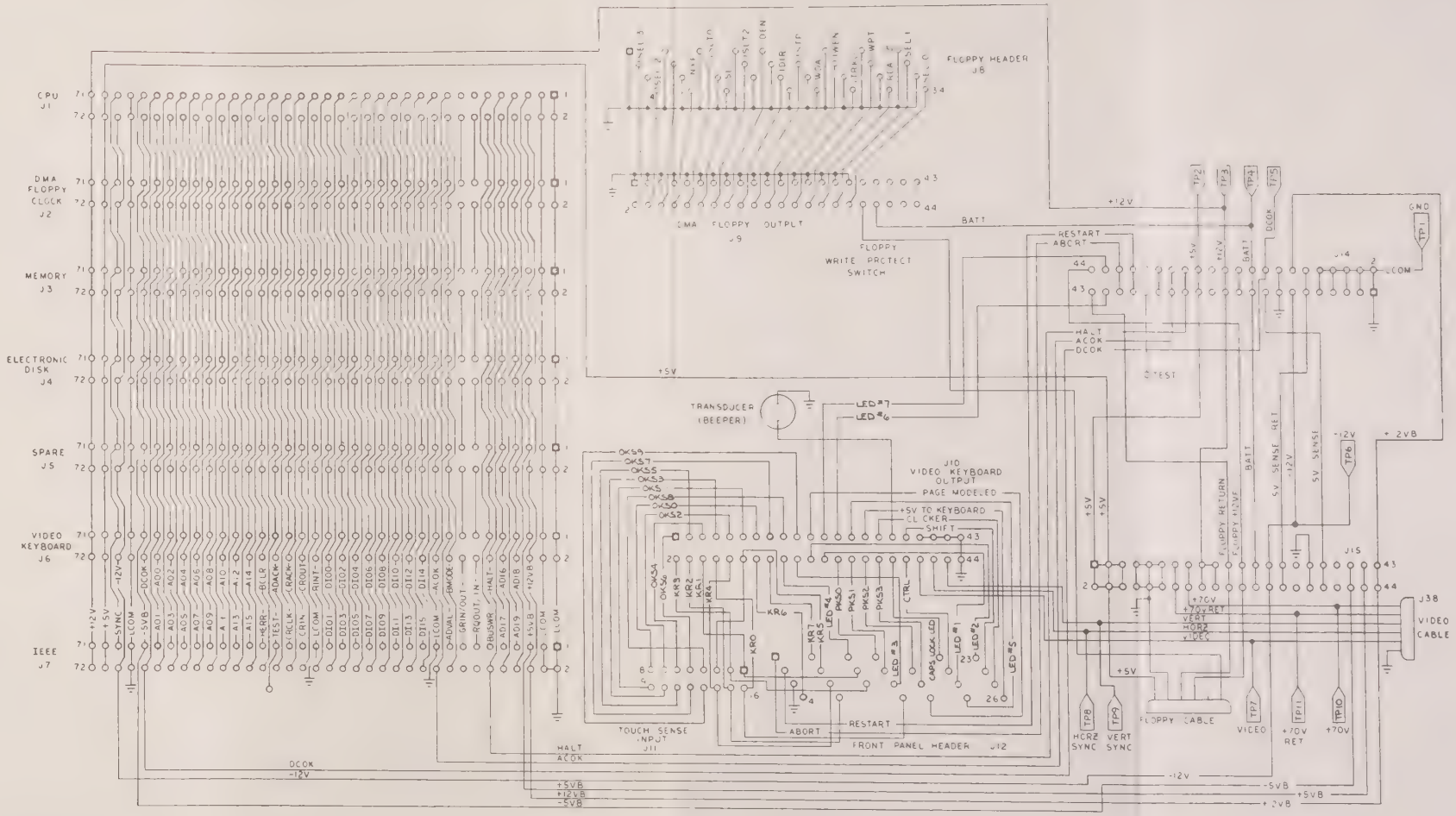


Figure 6-12. Mother Board Schematic

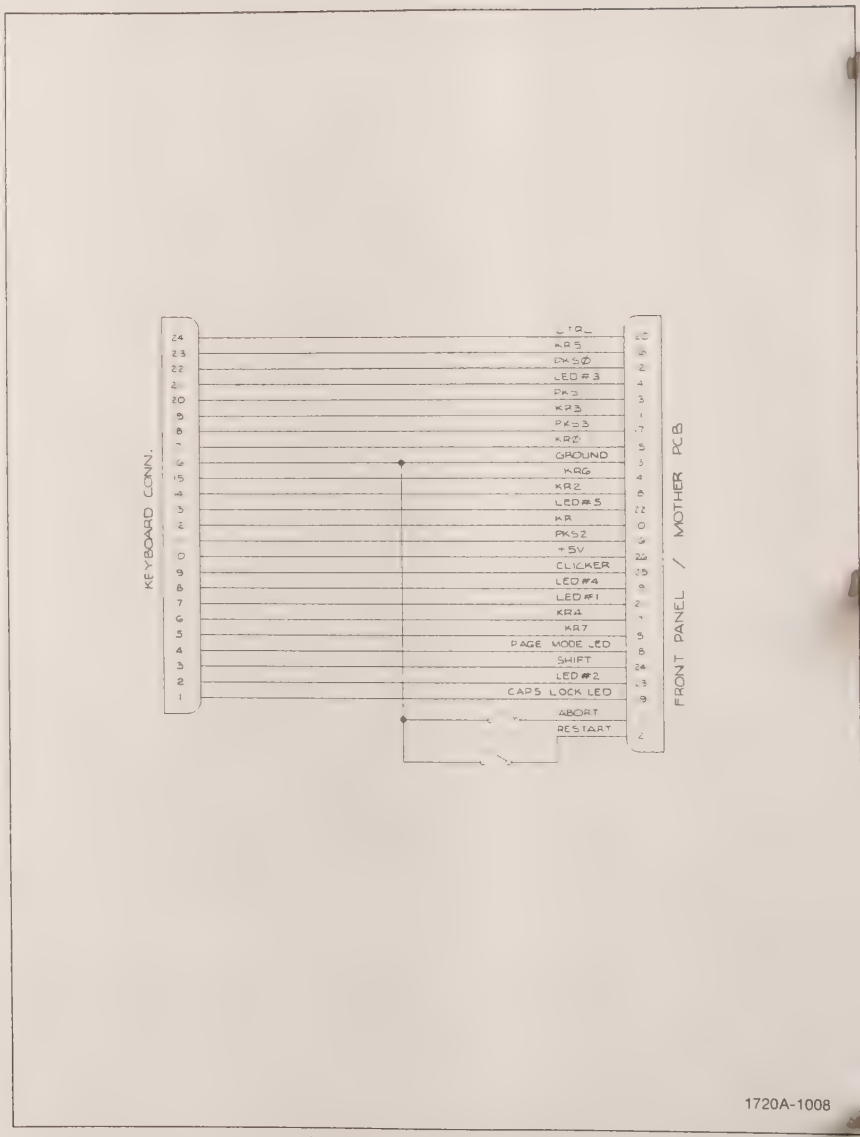
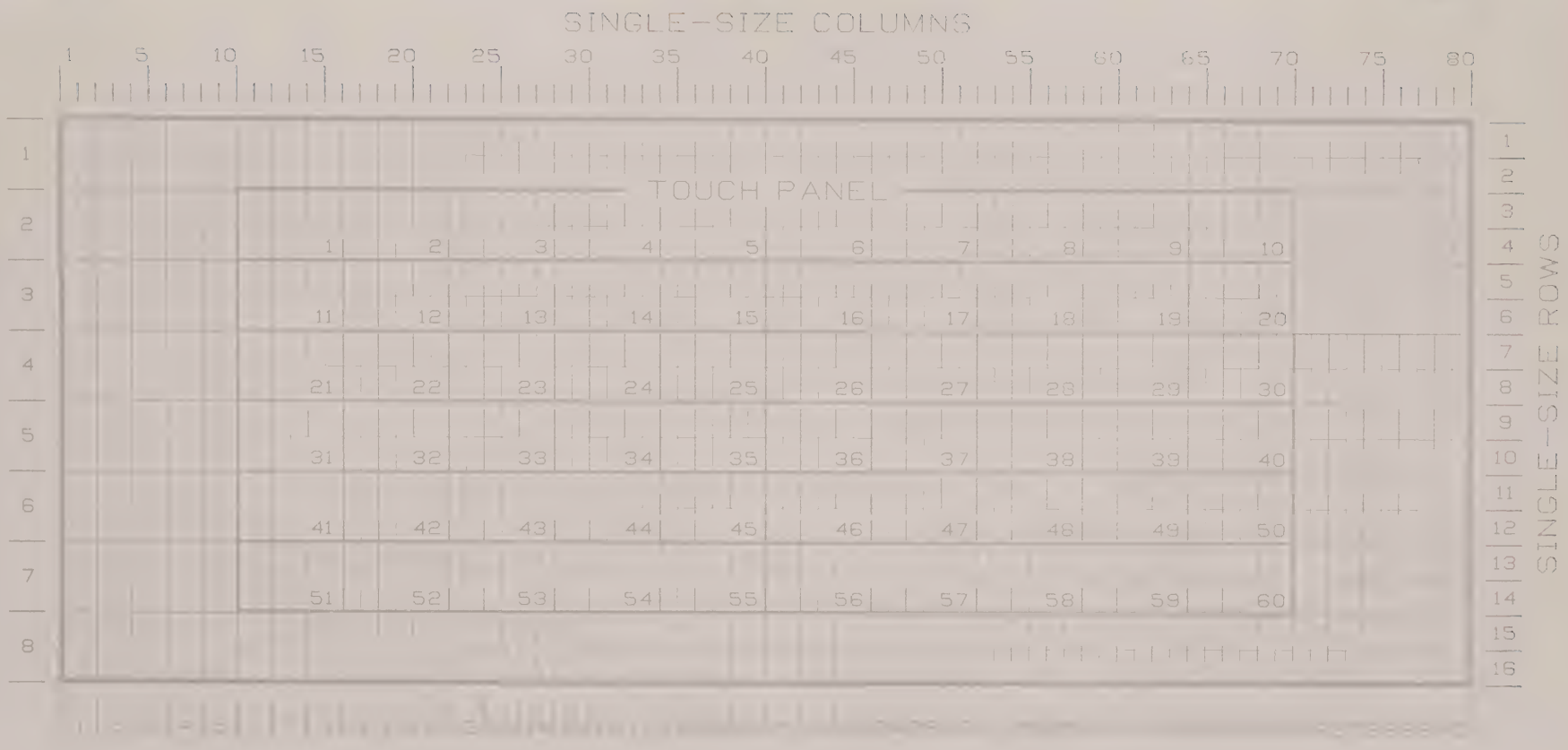


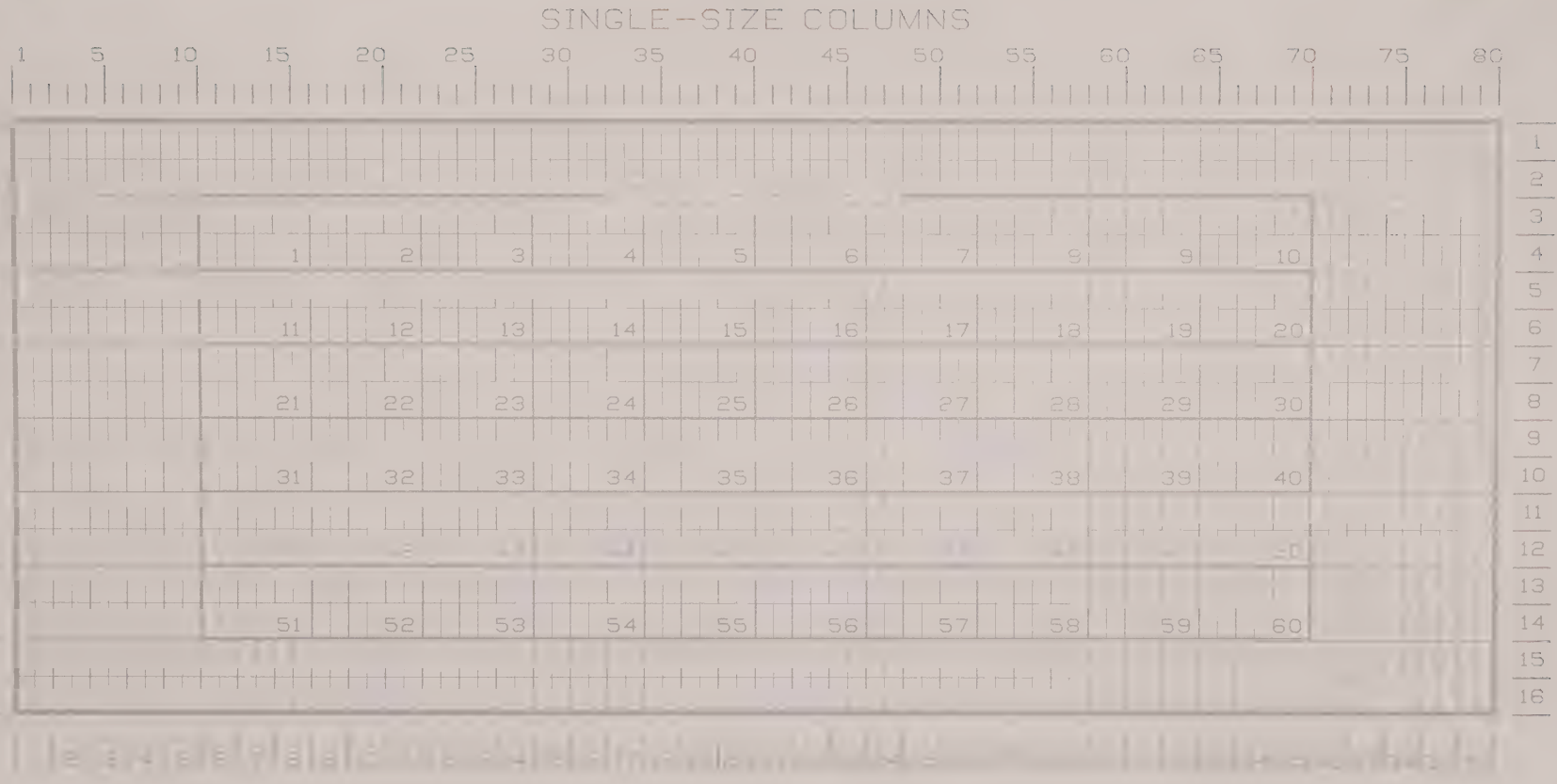
Figure 6-13. Front Panel Schematic



Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____



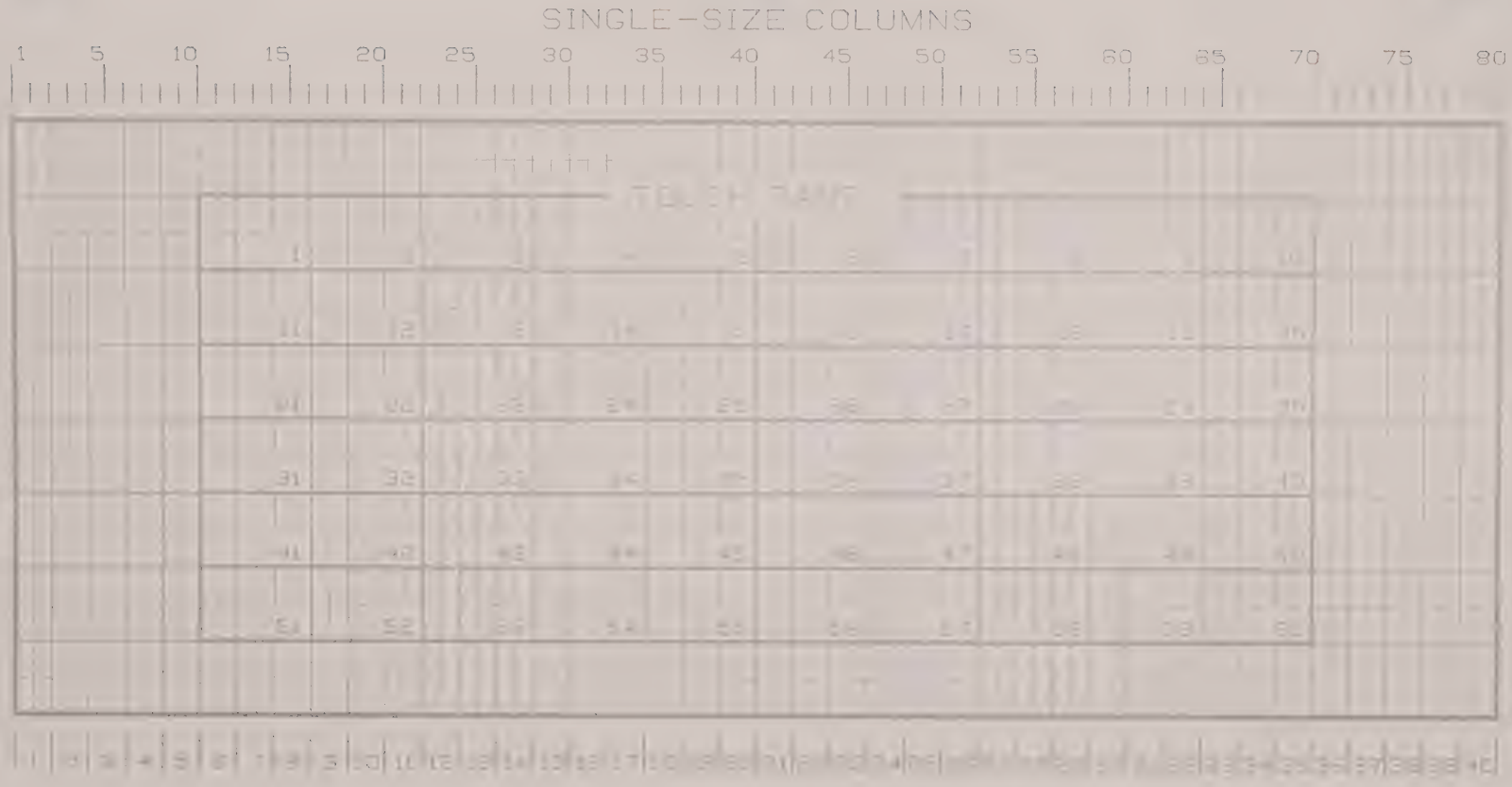
Notes:

Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ____ of ____

FLUKE

Reorder under Fluke P/N 533547 Rev 1 6 83 pad of 50



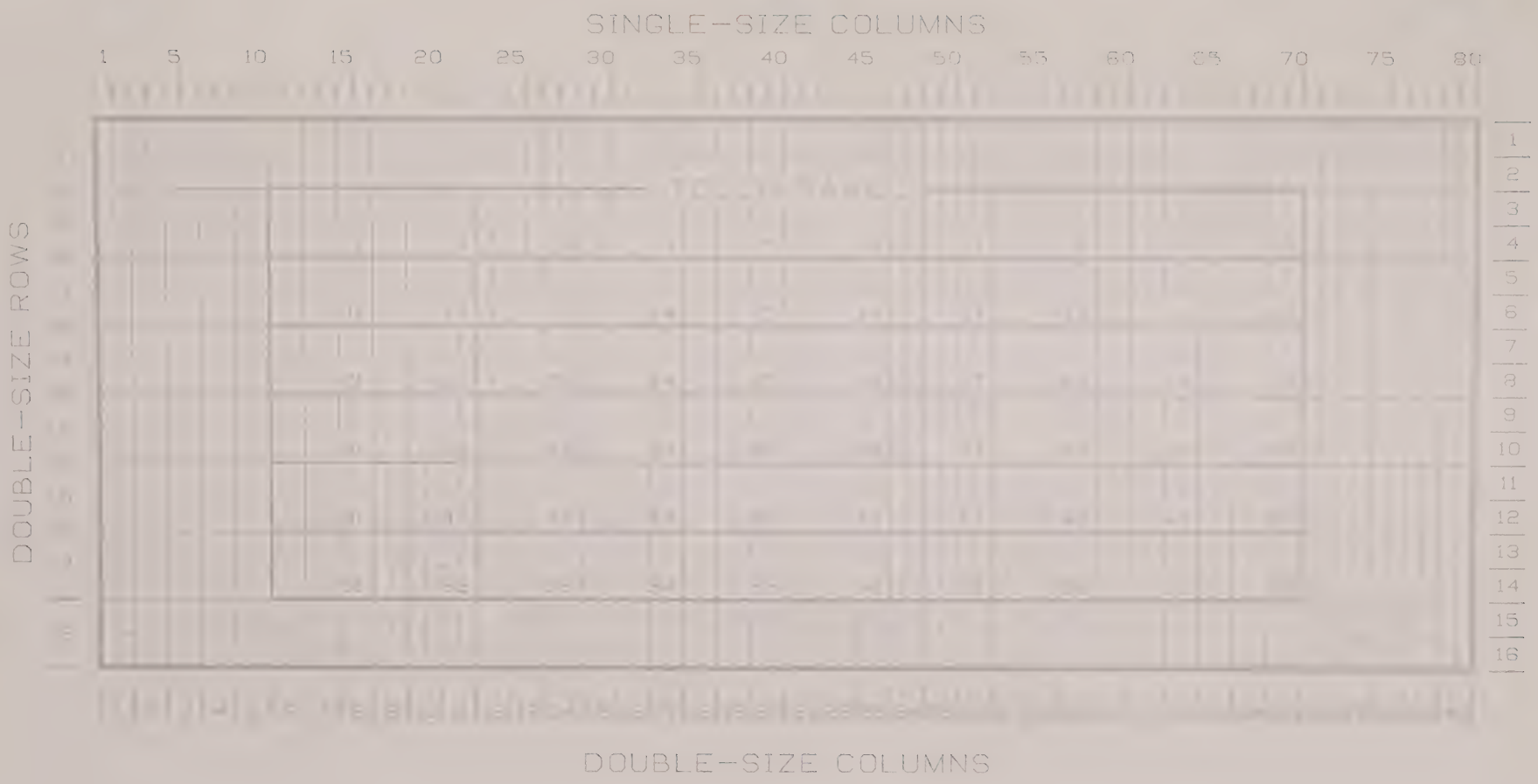
Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

FLUKE

Reorder under Fluke P/N 533547 Rev 1 6/83 (pad of 50)


SINGLE-SIZE COLUMNS
1 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80
DOUBLE-SIZE ROWS
SINGLE-SIZE ROWS
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
DOUBLE-SIZE COLUMNS

Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

FLUKE

SINGLE-SIZE COLUMNS

1 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80

DOUBLE-SIZE ROWS

SINGLE-SIZE ROWS

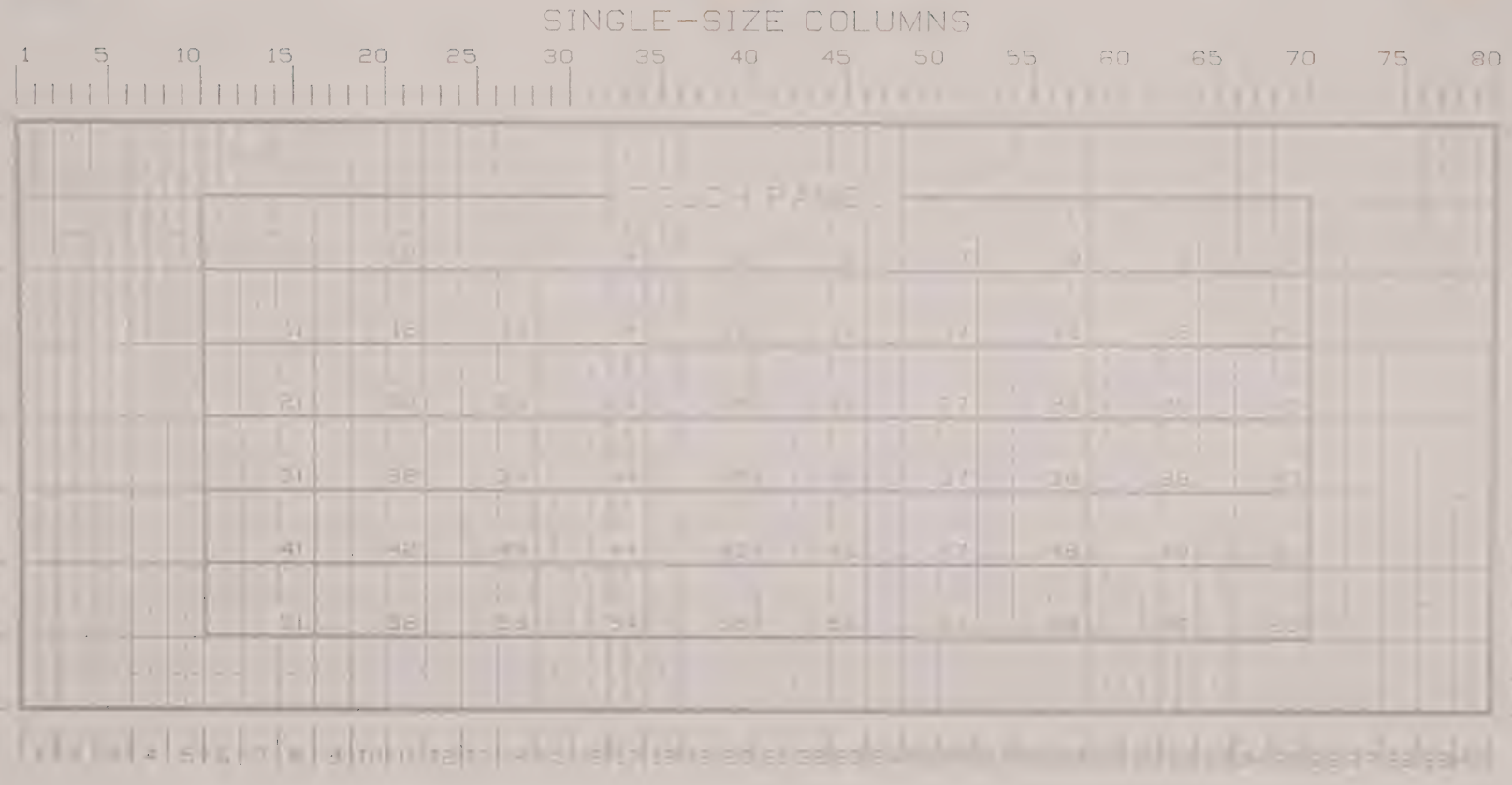
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16

Notes:

Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

FLUKE



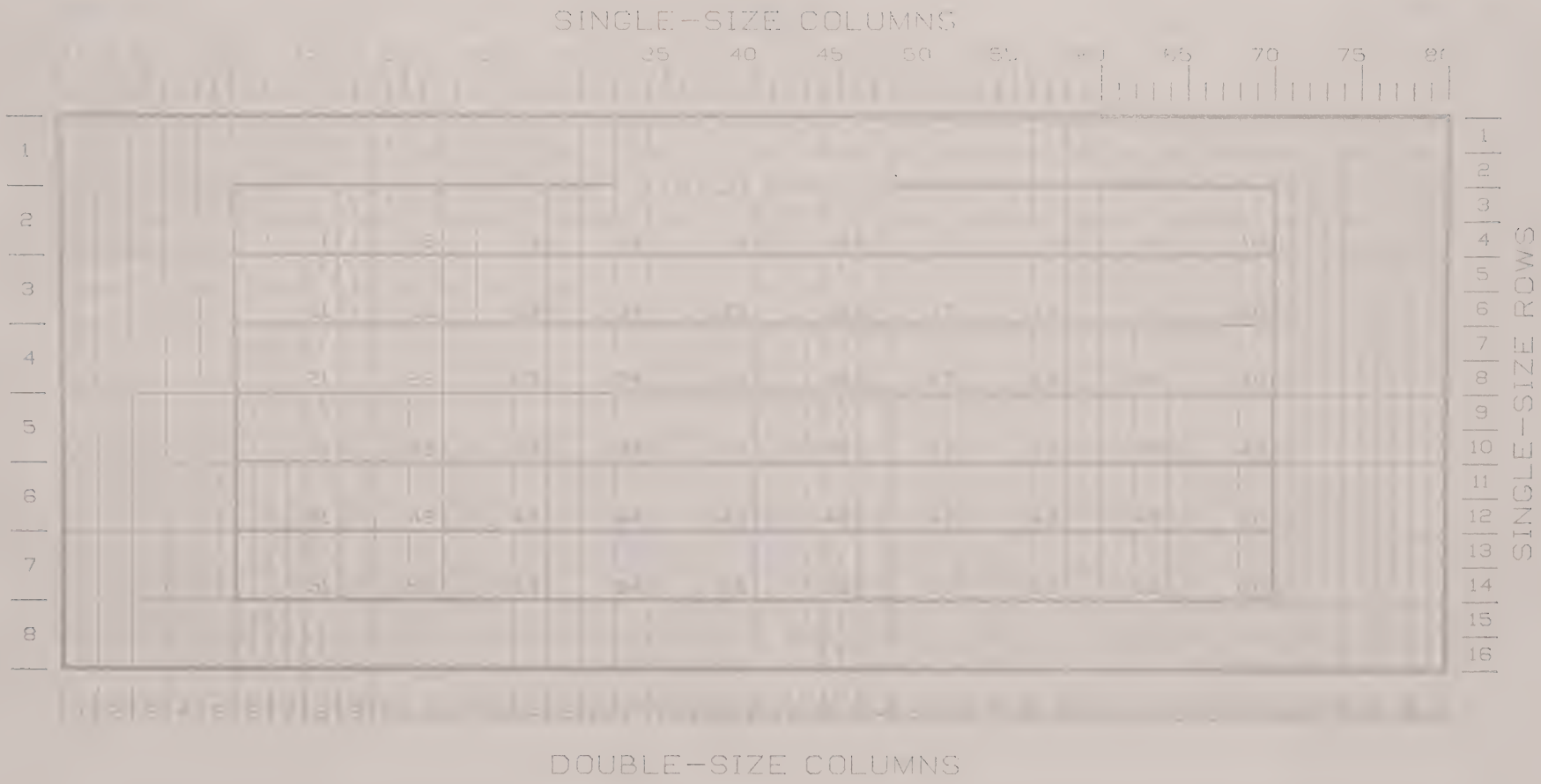
Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

FLUKE

Reorder under Fluke P/N 533547 Rev 1 6/83 (pad of 50)



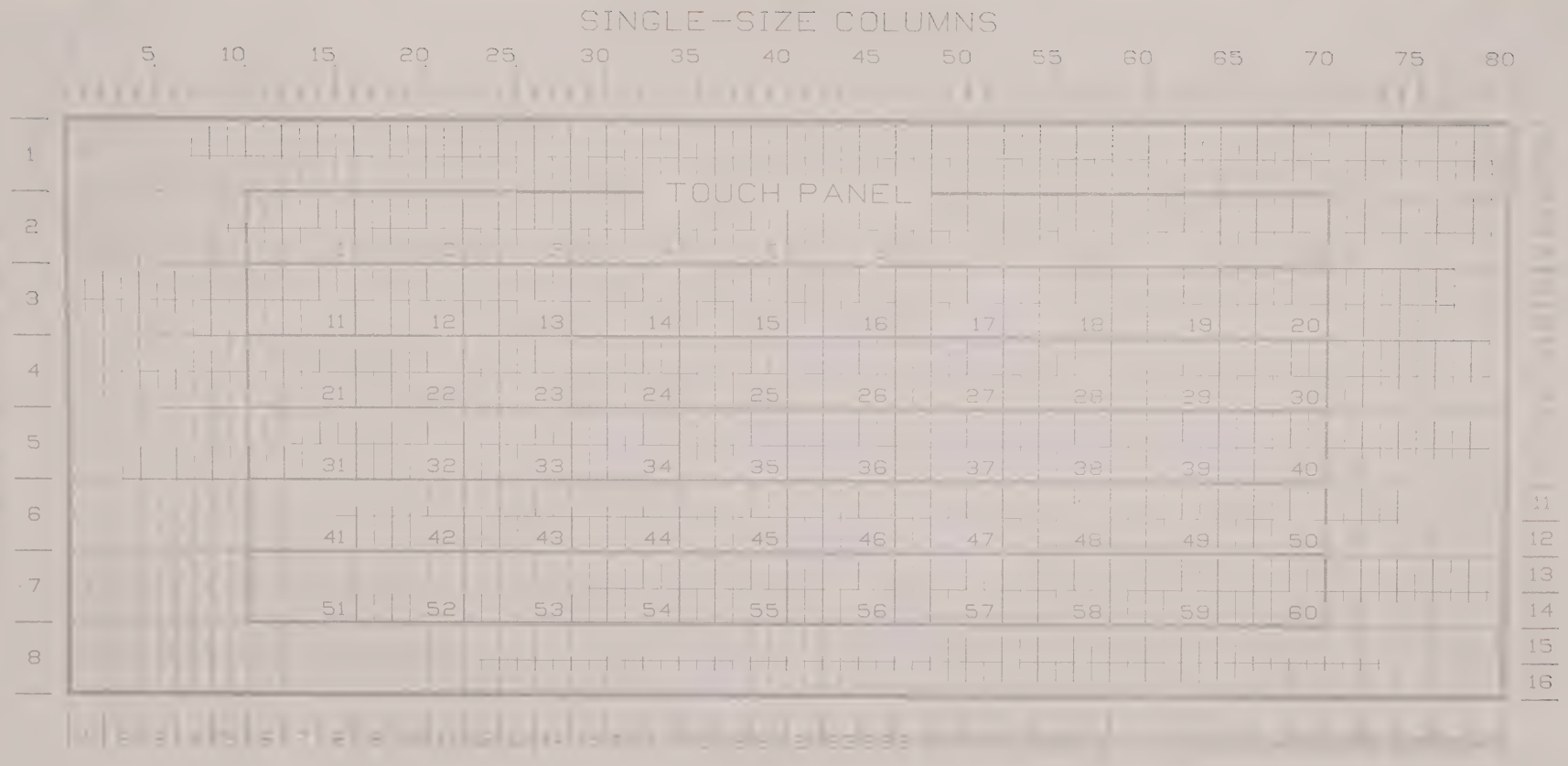
Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

FLUKE

Fluke P/N 833547 Rev 1 6/93 pad of 50



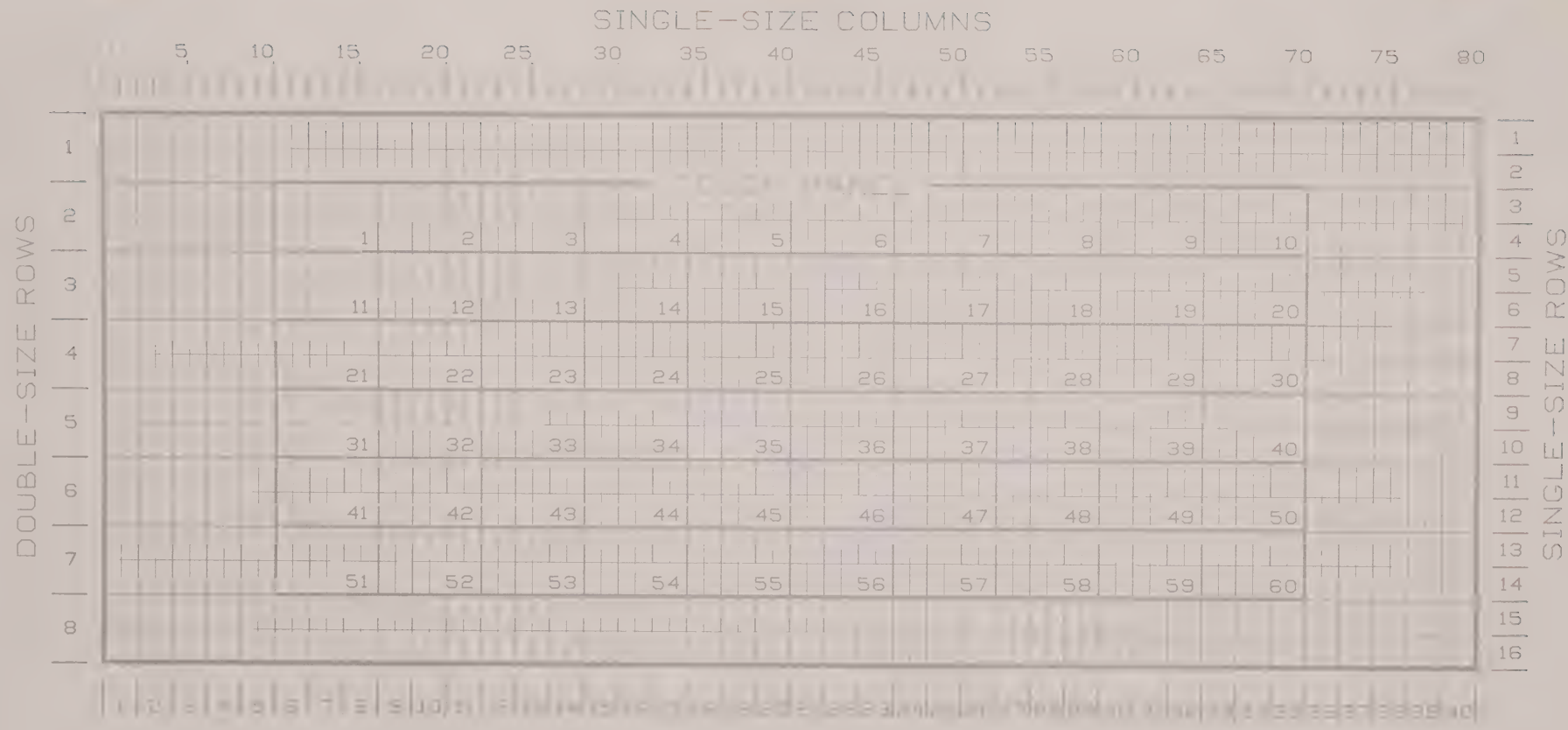
Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____

FLUKE

Reorder under Fluke P/N 533847 Rev 1 6/83 pad of 50



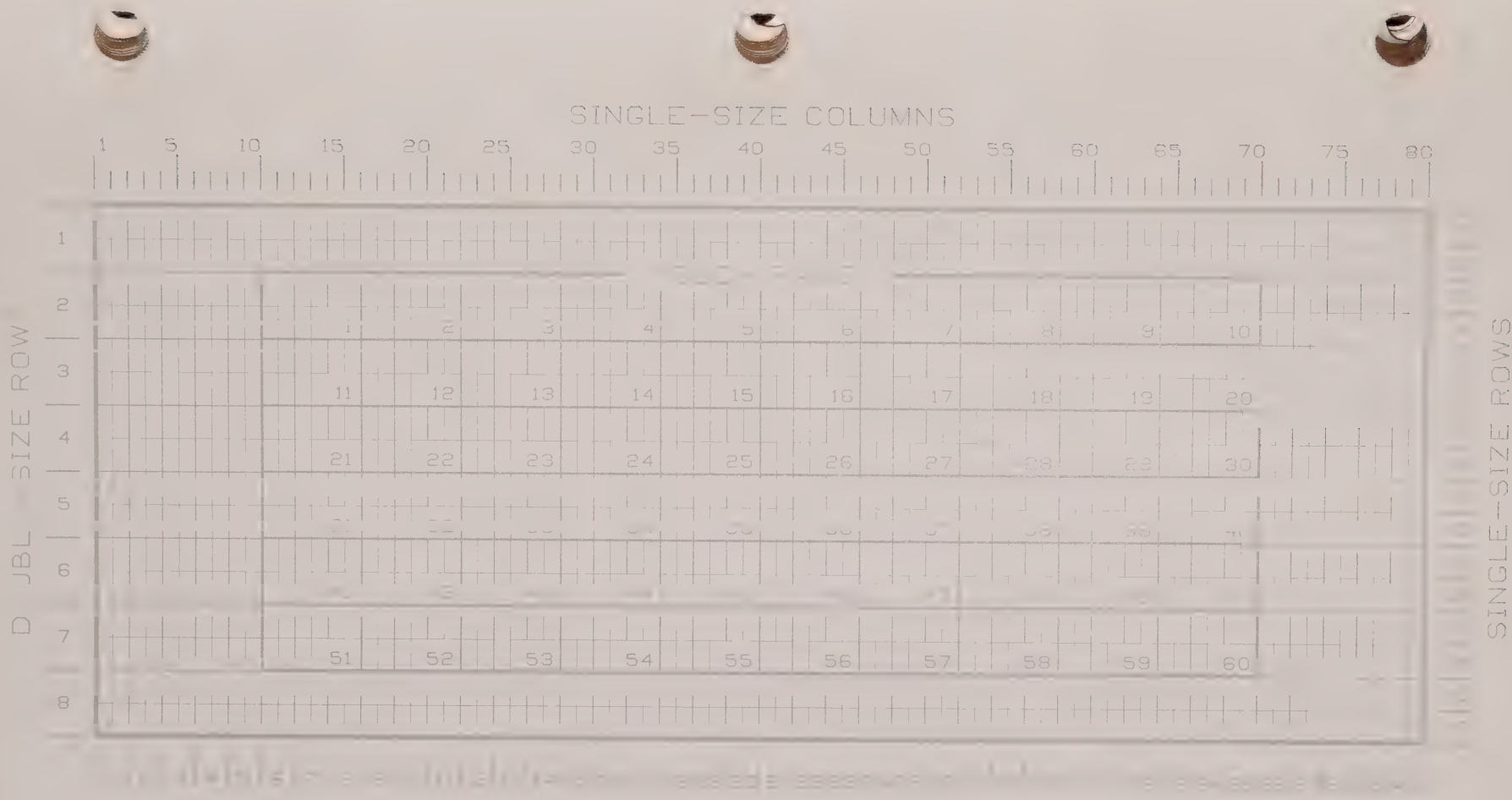
Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

FLUKE

Reorder under Fluke P/N 533547 Rev 1 6/83 (pad of 50)



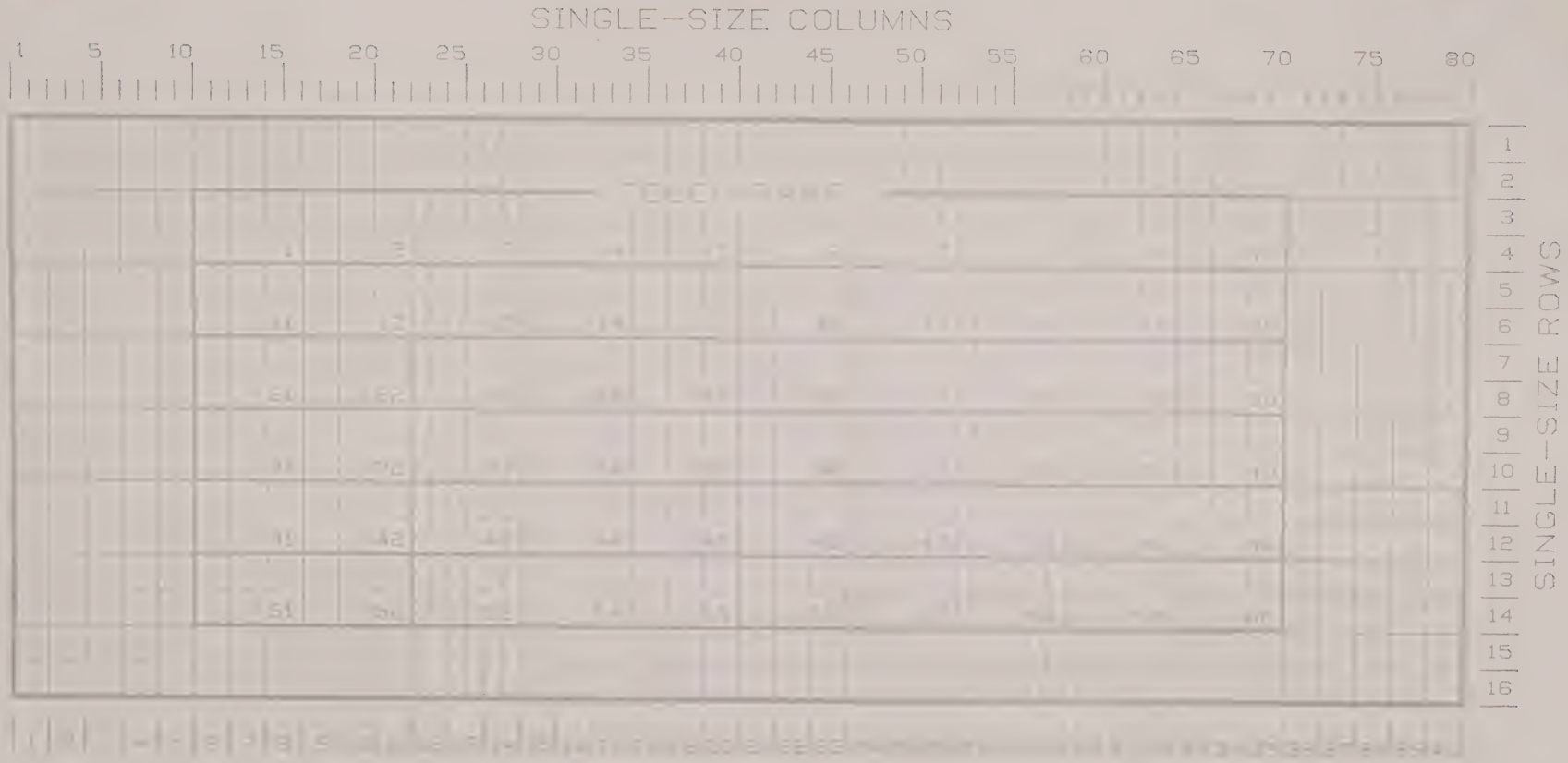
Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____

Printed under Fluke P/N 622267 Rev. 1 6/83 (1 out of 50)

FLUKE


SINGLE-SIZE COLUMNS
1 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80
DOUBLE-SIZE ROWS
SINGLE-SIZE ROWS
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
DOUBLE-SIZE COLUMNS

Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

FLUKE

Reorder under Fluke P/N 533547 Rev 1 6/82 (pad of 50)

Notes: _____

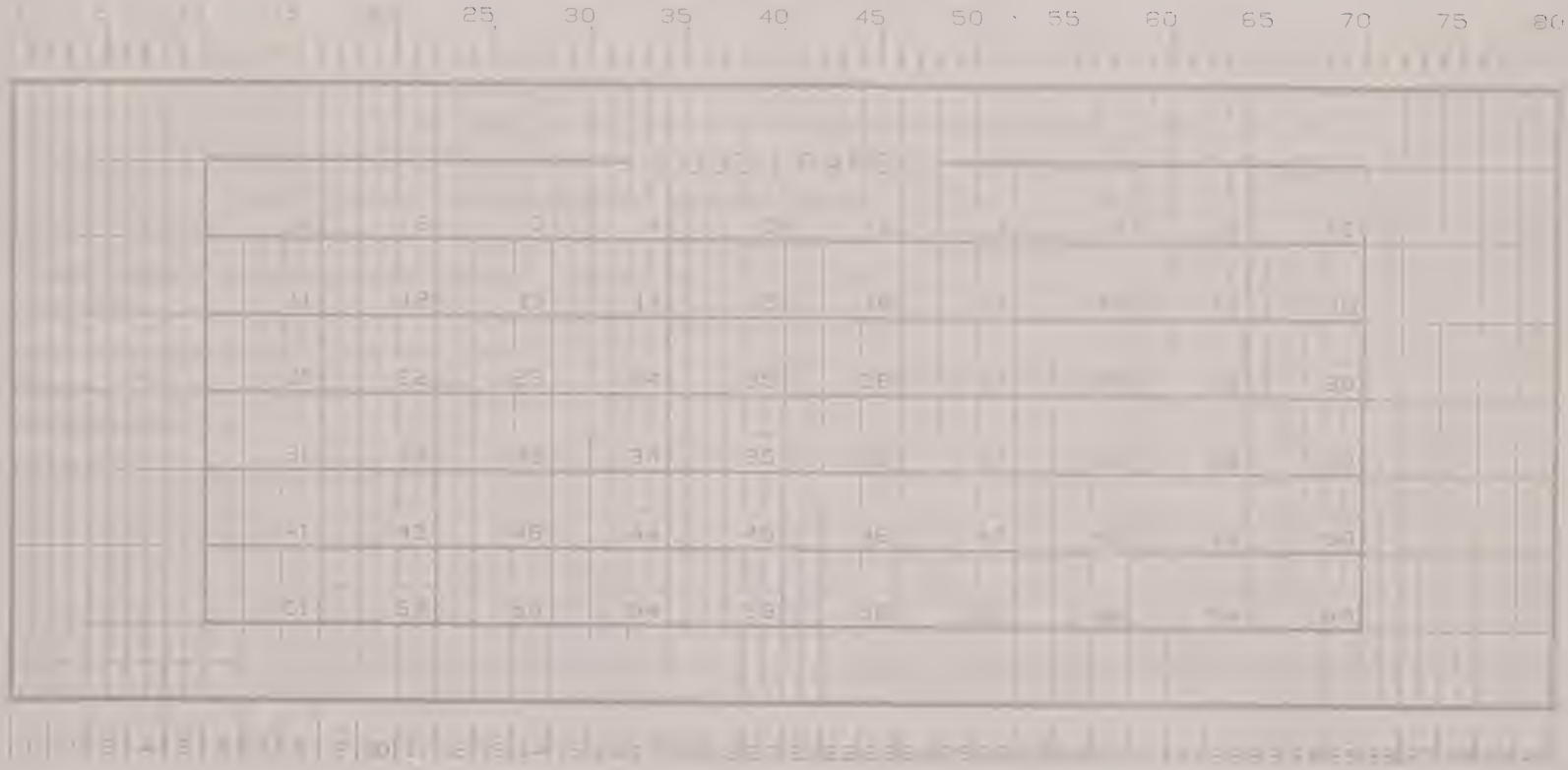
Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

FLUKE

Reorder under Fluke P/N 533547 Rev 1 6/83 (pad of 50)

SINGLE–SIZE COLUMNS



DOUBLE–SIZE ROWS

SINGLE–SIZE ROWS

DOUBLE–SIZE COLUMNS

Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ____ of ____

FLUKE

Reorder under Fluke P/N 533547 Rev 1 6/83 (pad of 50)

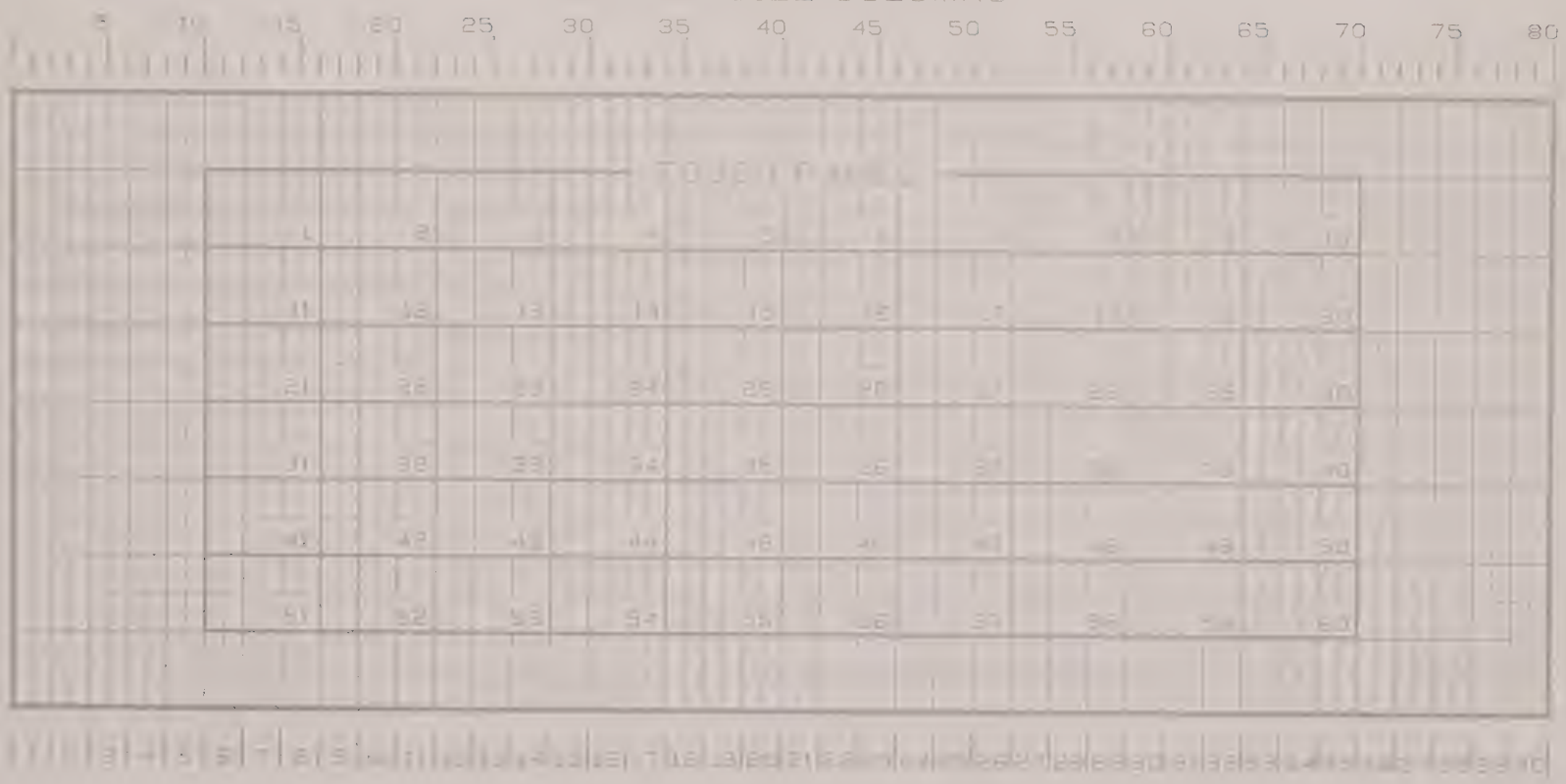
Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

FLUKE

Reorder under Fluke P/N 532547 Rev 1 5/83 (pad of 50)

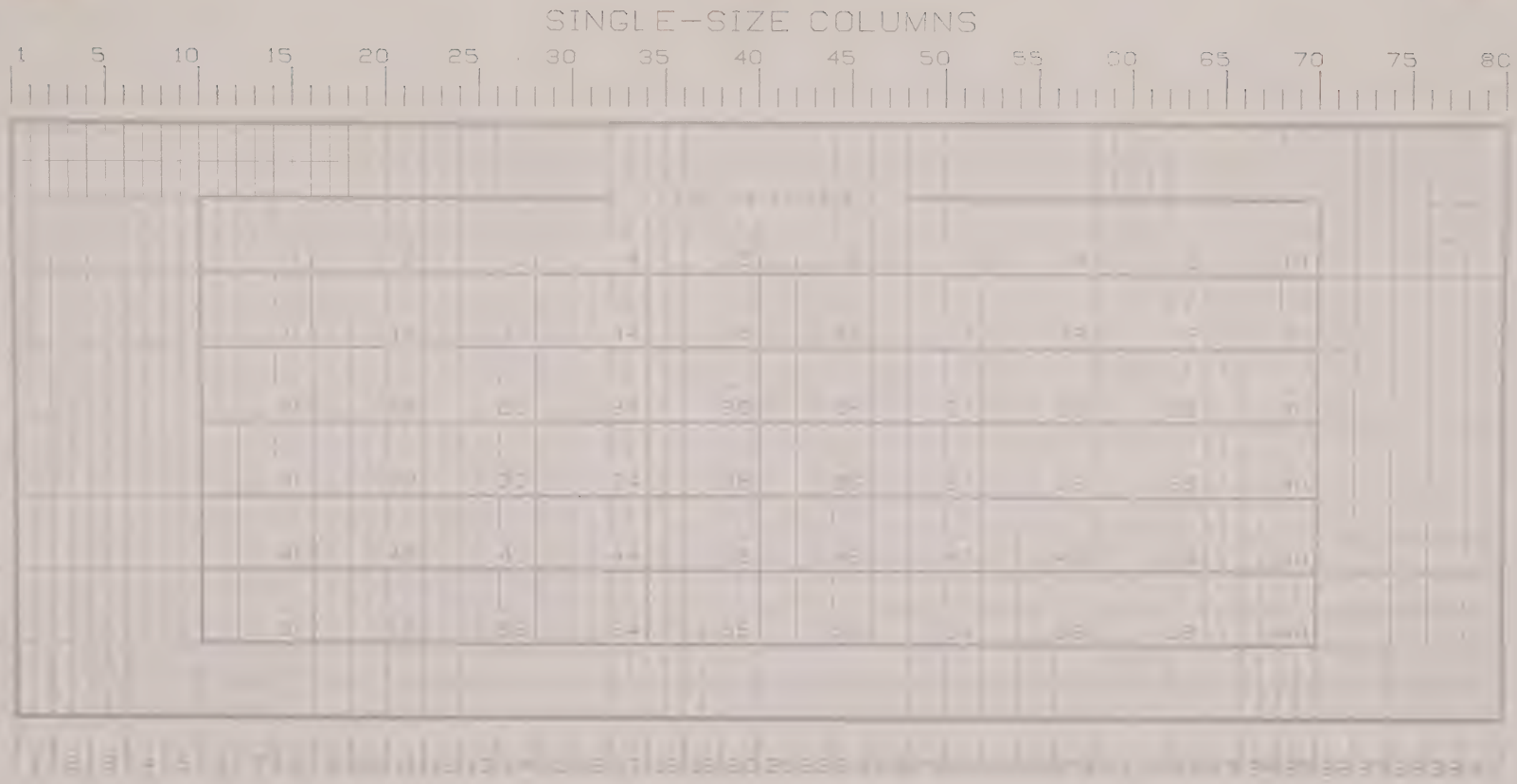

SINGLE–SIZE COLUMNS
5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80
DOUBLE–SIZE ROWS
SINGLE–SIZE ROWS
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
DOUBLE–SIZE COLUMNS

Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____

Reorder under Fluke P/N 533547 Rev 1 6/83 (pad of 50)

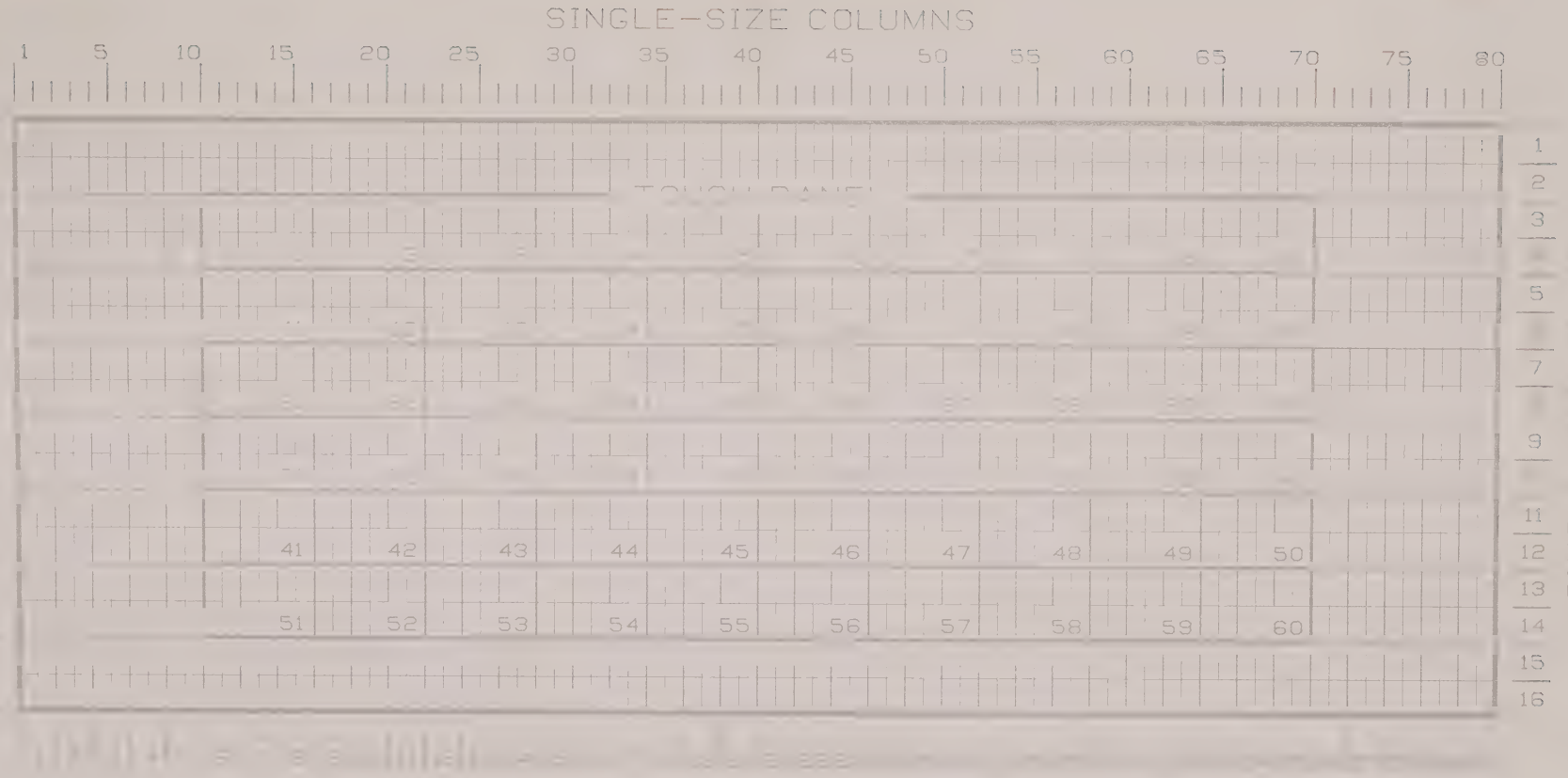

SINGLE-SIZE COLUMNS
1 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80
DOUBLE-SIZE ROWS
SINGLE-SIZE ROWS
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
DOUBLE-SIZE COLUMNS

Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____

Vendor Number Fluke P/N 533804 / Rev. 1 6/83 (pad of 50)



Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____

Reorder under Fluke P/N 533547 Rev 1 6/83 (pad of 50)

SINGLE-SIZE COLUMNS

DOUBLE-SIZE ROWS

SINGLE-SIZE ROWS

DOUBLE-SIZE COLUMNS

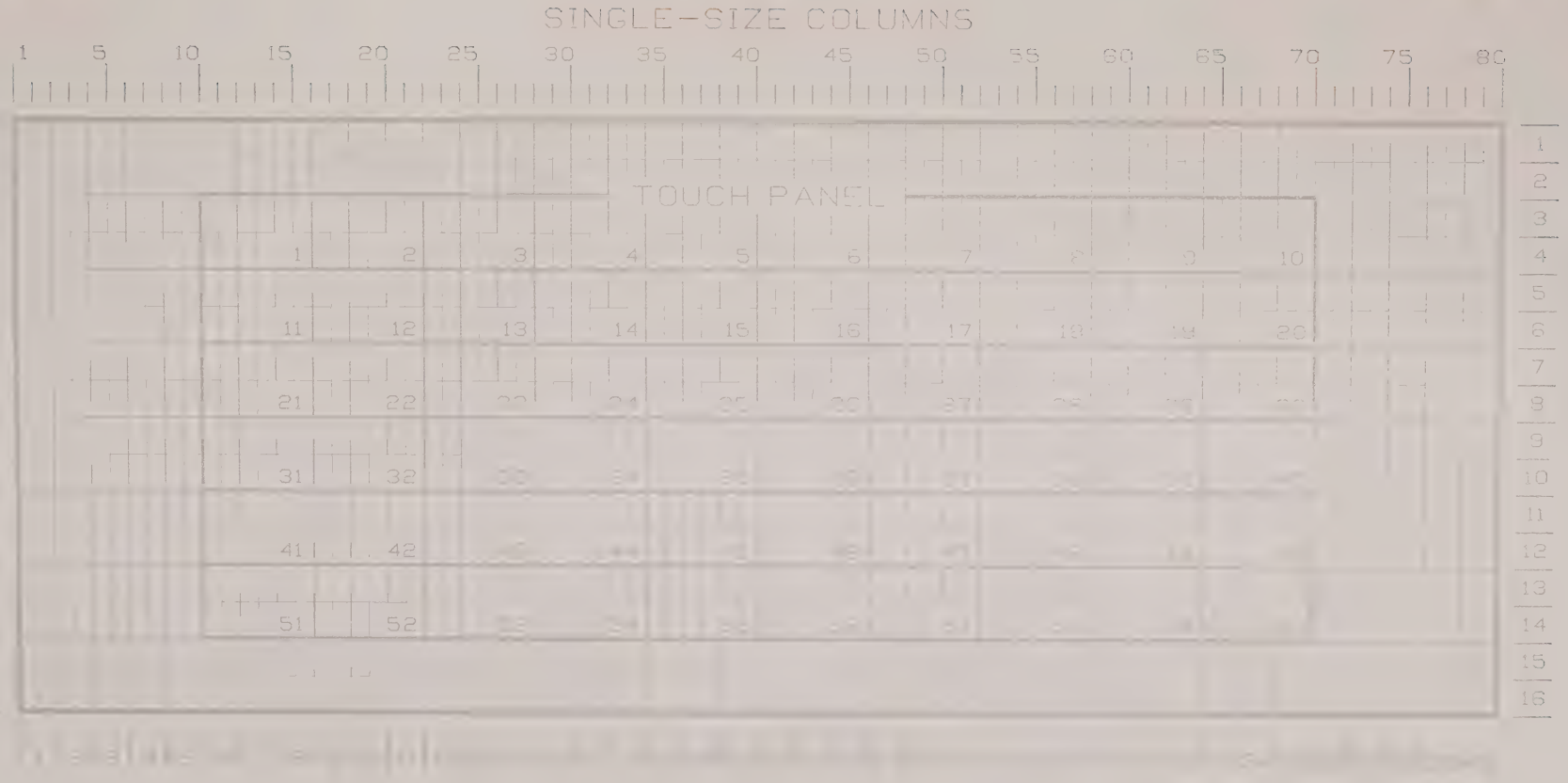
Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

FLUKE

Reorder under Fluke P/N 533547 Rev 1 6/83 (pad of 50)



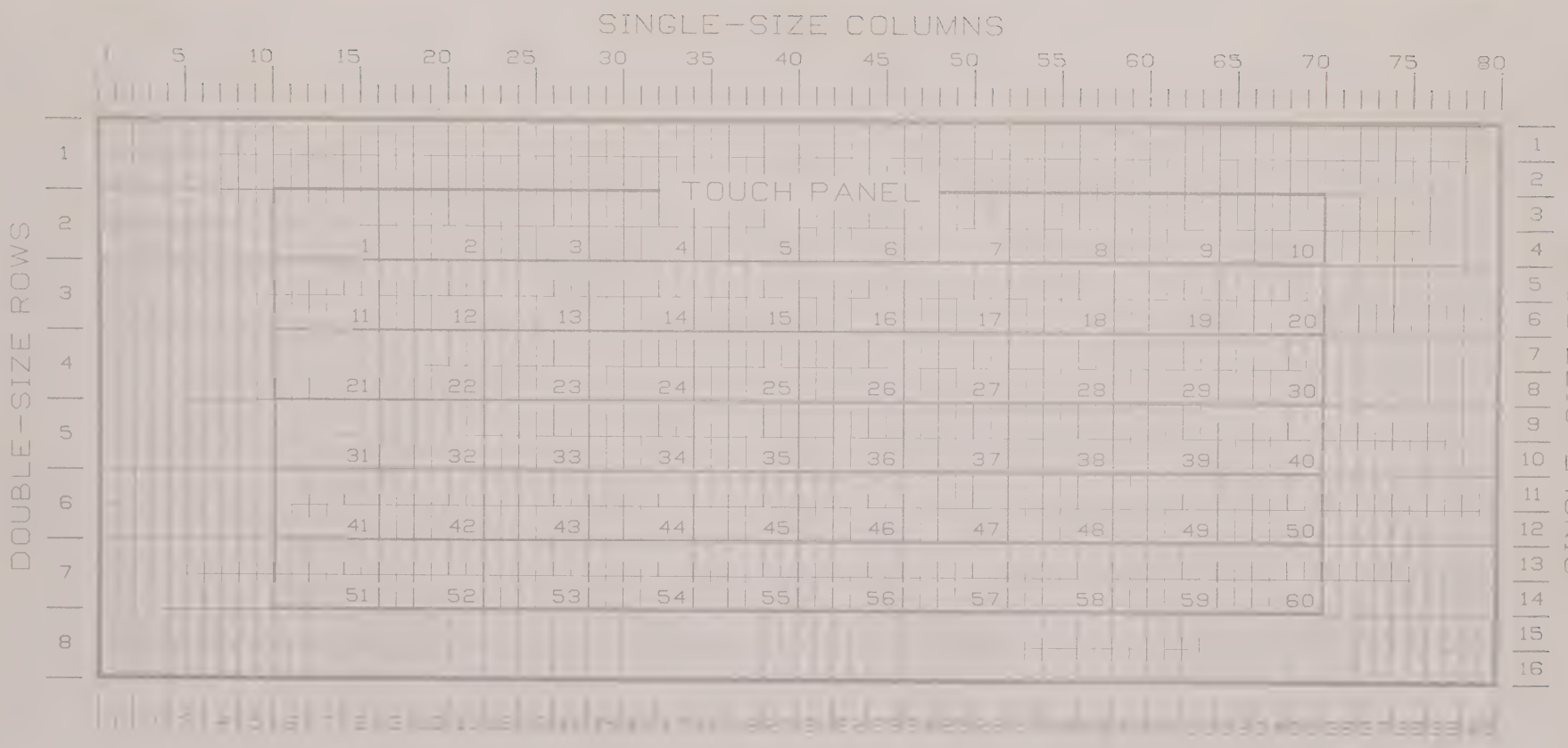
Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ____ of ____

FLUKE

Reorder Info: Fluke P/N 533562 Rev 1 6/92 (part 1 of 2)



Notes: _____

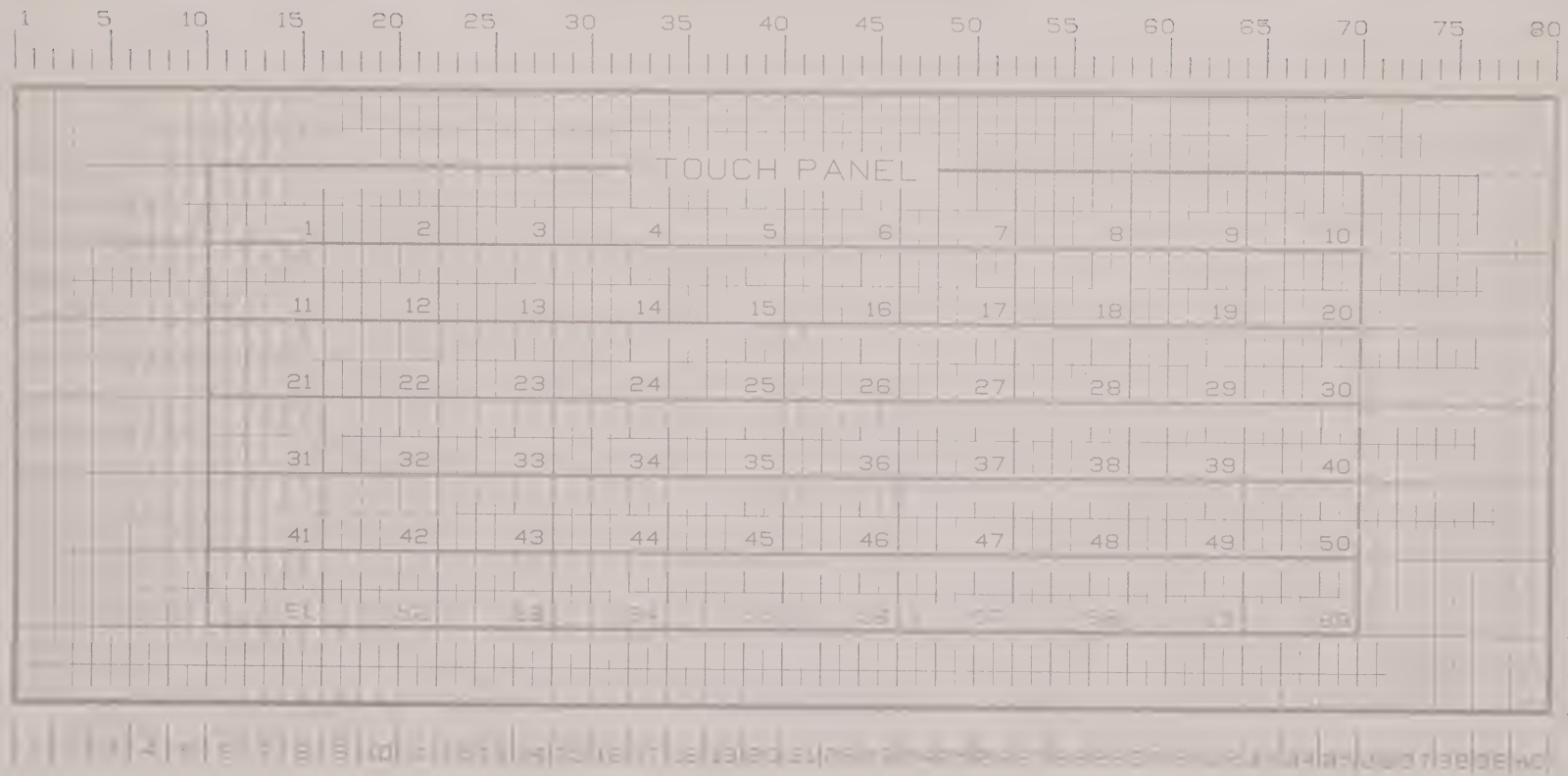
Controller Programming Work Sheet

_____ Programmer _____ Page ____ of ____

FLUKE

Reorder under Fluke P/N 533547 Rev 1 6/83 (pad of 50)

SINGLE–SIZE COLUMNS



DOUBLE–SIZE ROWS

SINGLE–SIZE ROWS

DOUBLE–SIZE COLUMNS

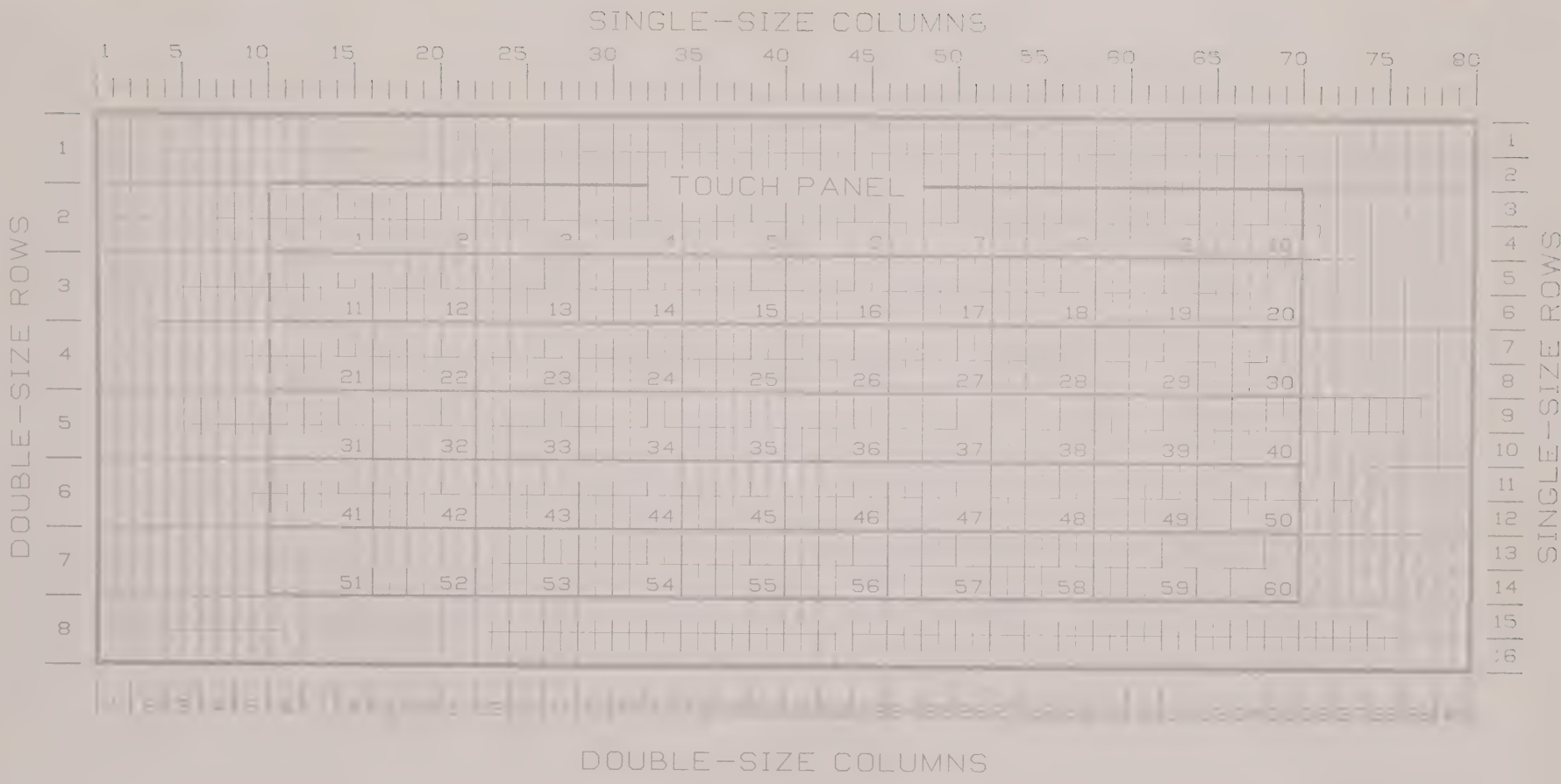
Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

FLUKE

Reorder under Fluke P/N 533547 Rev 1 6/83 (pad of 50)



Notes: _____

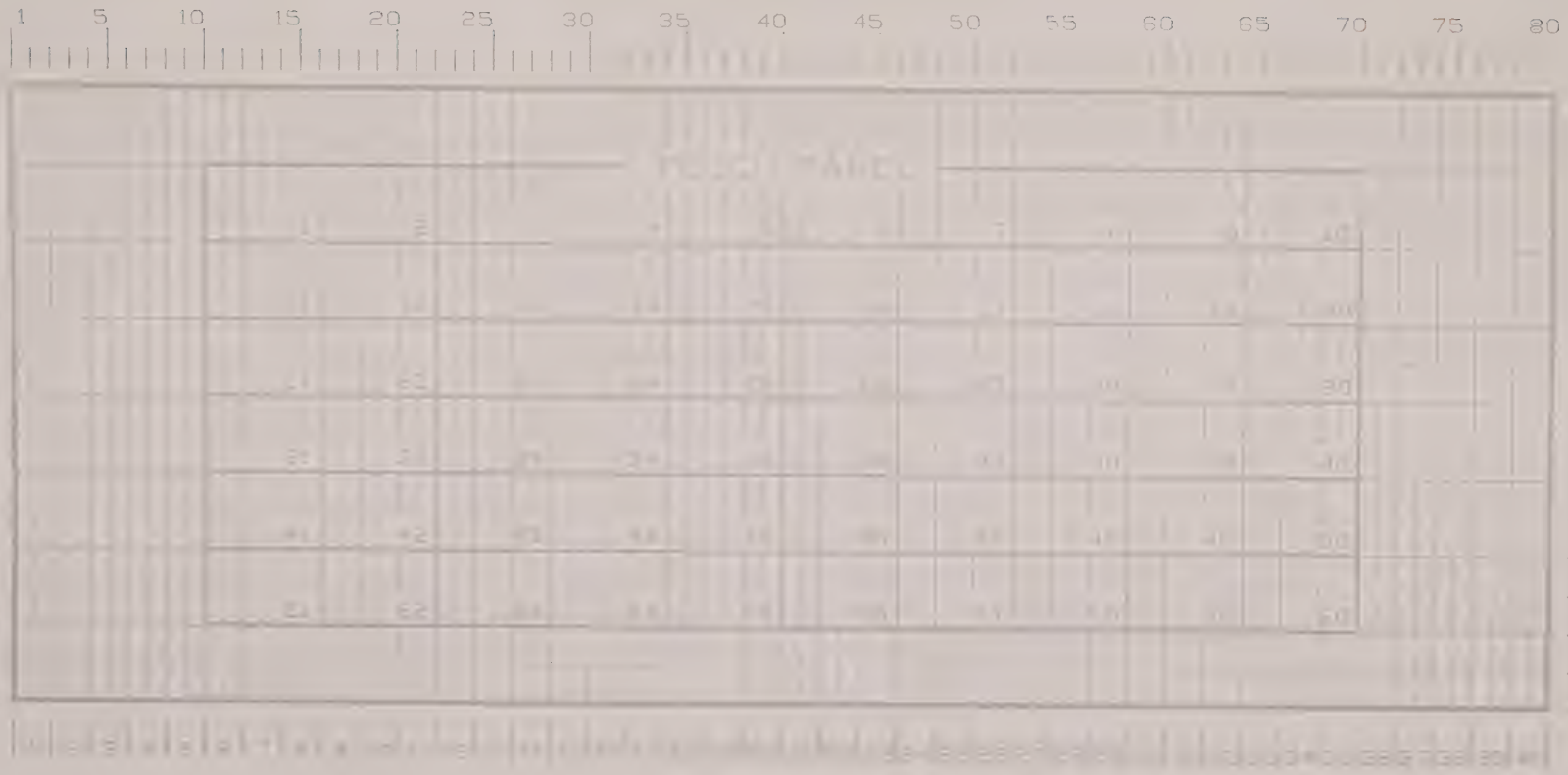
Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

FLUKE

Reorder under Fluke P/N 833547 Rev. 1 6/83 (pad of 50)

SINGLE-SIZE COLUMNS



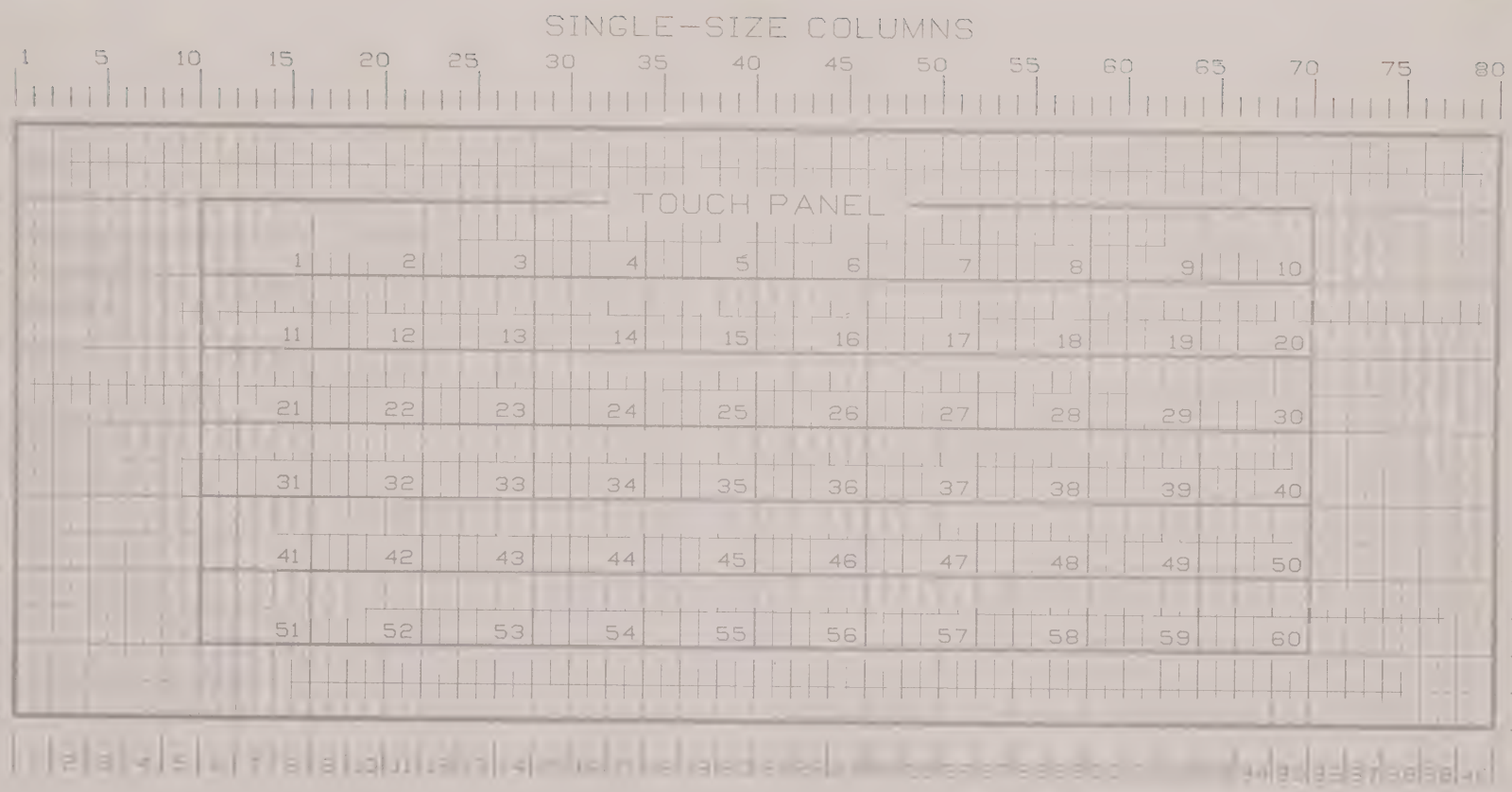
DOUBLE-SIZE COLUMNS

Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____

Reorder under Fluke P/N 533547 Rev 1 6/83 (pad of 50

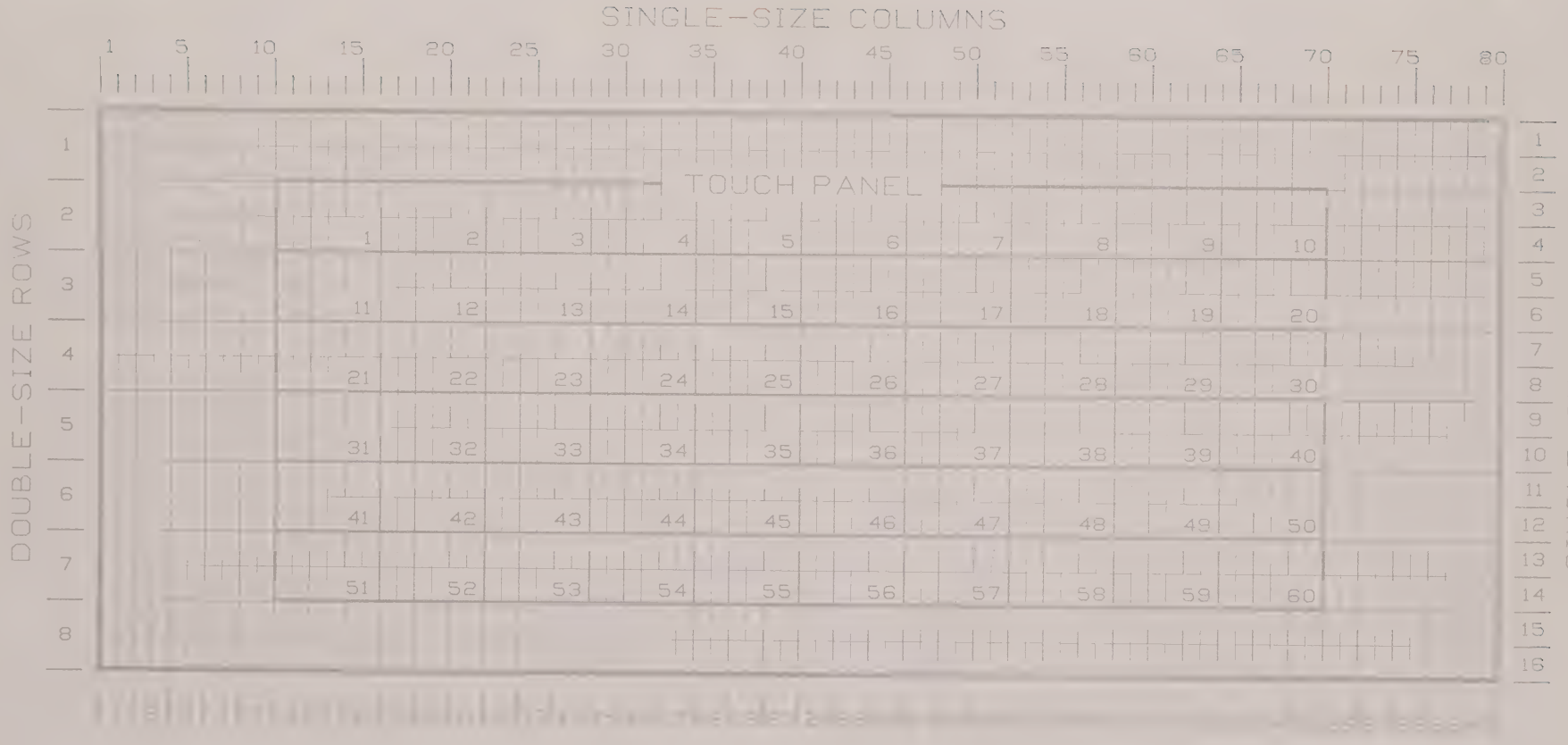


Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____

Reorder under Fluke P/N 533547 Rev 1 6/83 (pad of 50)



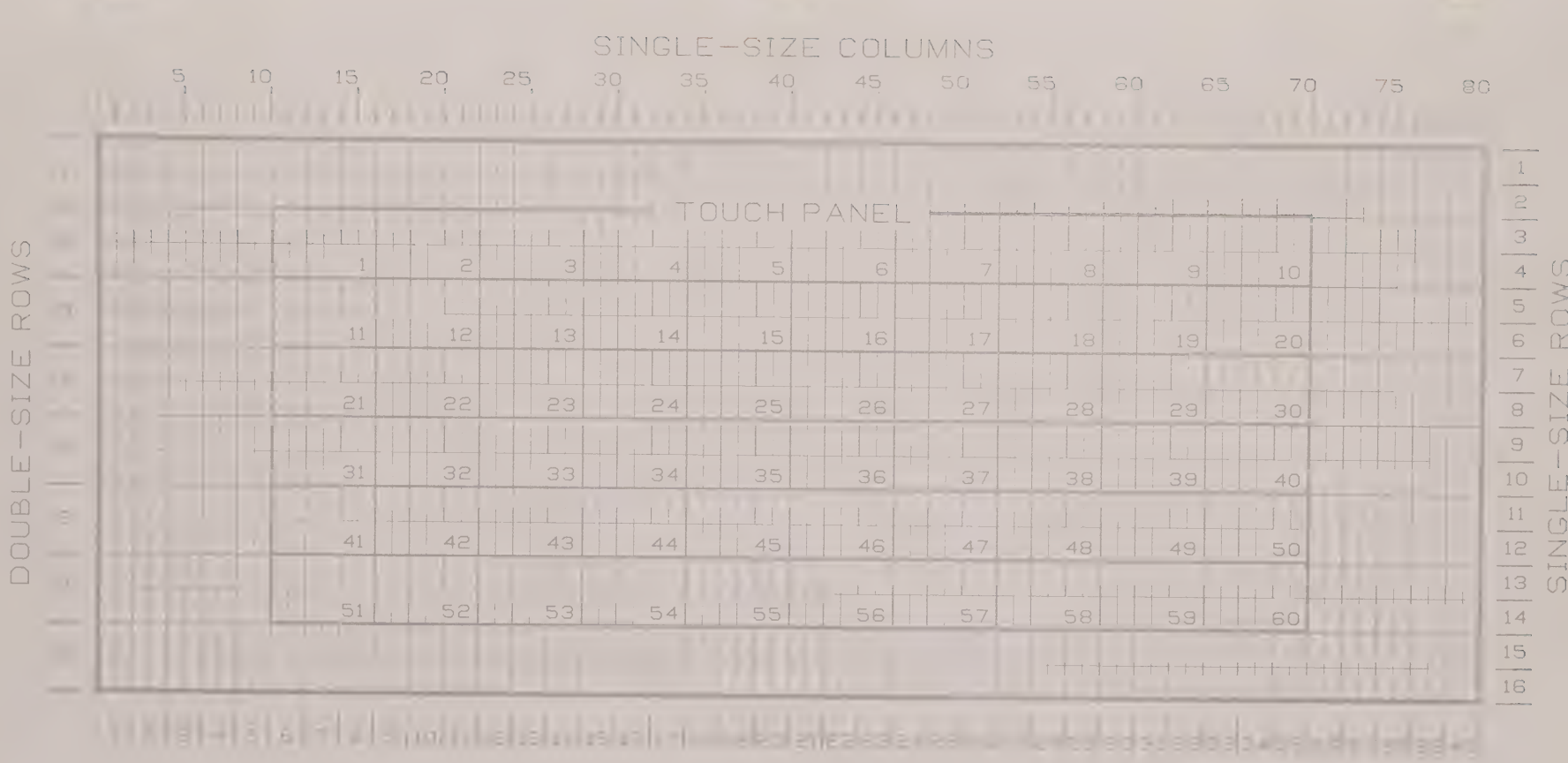
Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

FLUKE

Reorder Jumer Fluke P/N 533047 Rev 1 6/83 (one of 50)



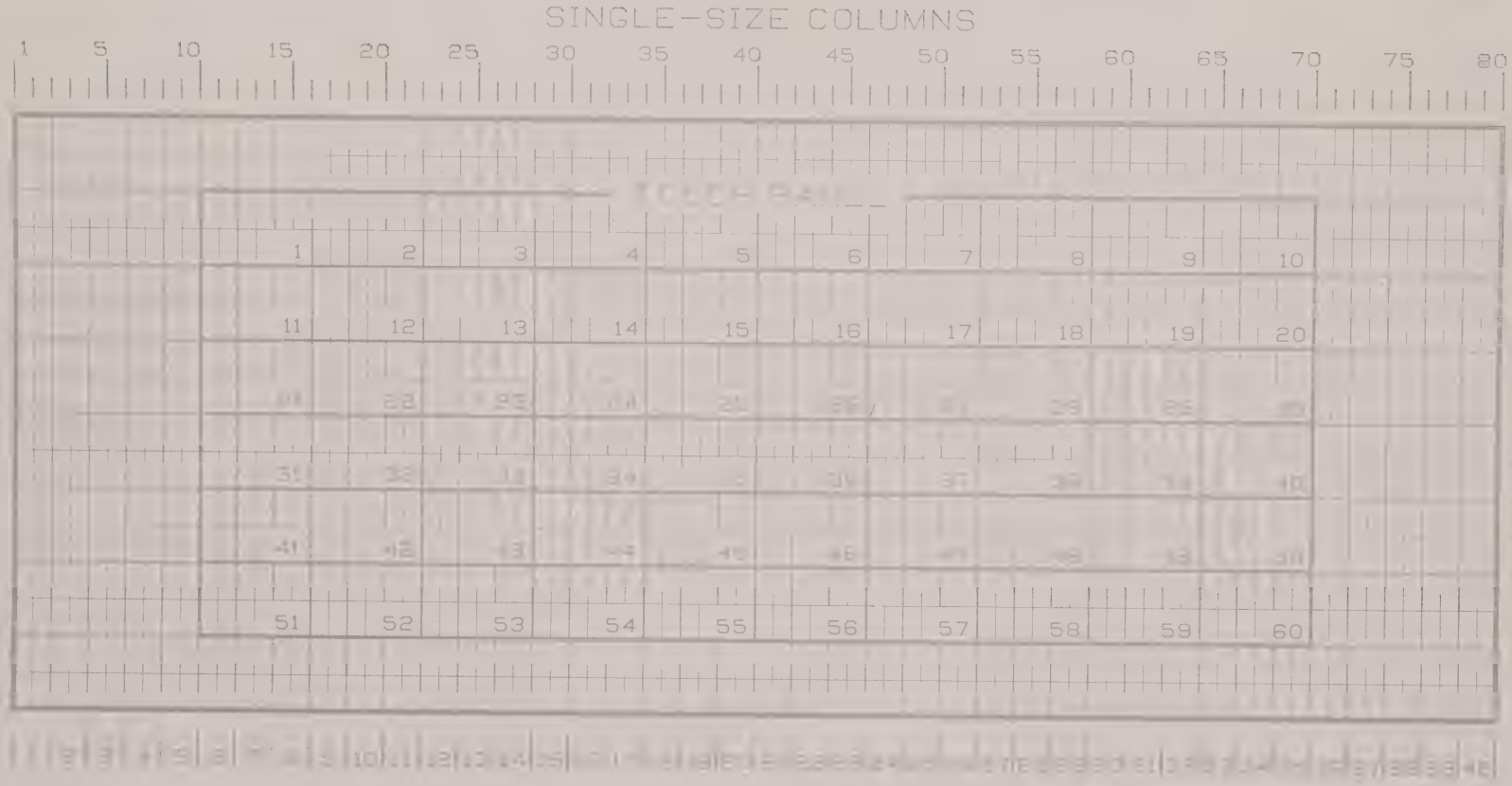
Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____ Rev. _____

FLUKE

Reorder under Fluke P/N 533547 Rev 1 6/83 (pad of 50)



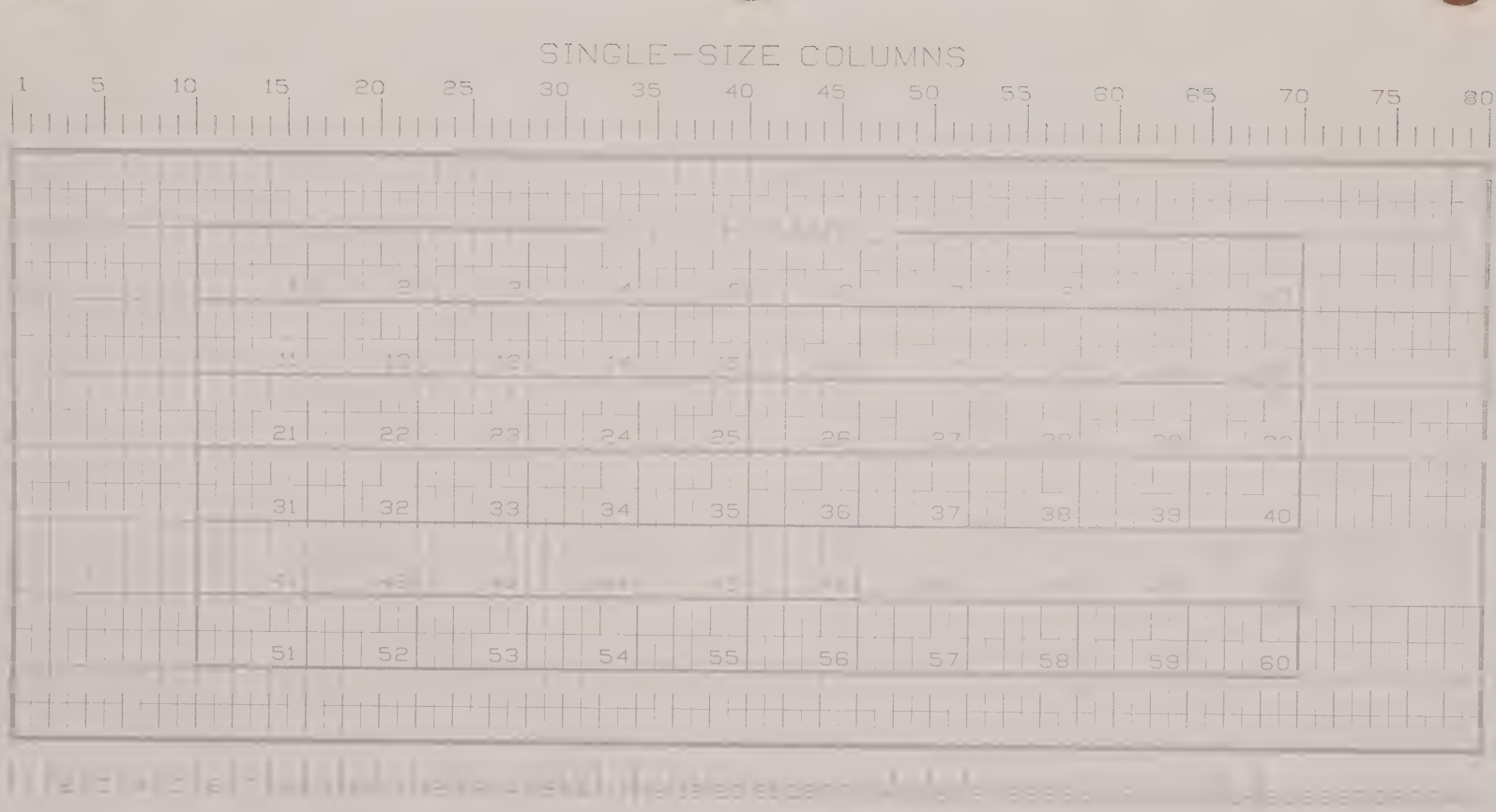
Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

FLUKE

Reorder under Fluke P/N 533547 Rev 1 6/83 (pad of 50)



Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

Reorder under Fluke P/N 533547 Rev 1 6/83 (pad of 50)

SINGLE-SIZE COLUMNS
1 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80
DOUBLE-SIZE ROWS
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
SINGLE-SIZE ROWS
DOUBLE-SIZE COLUMNS

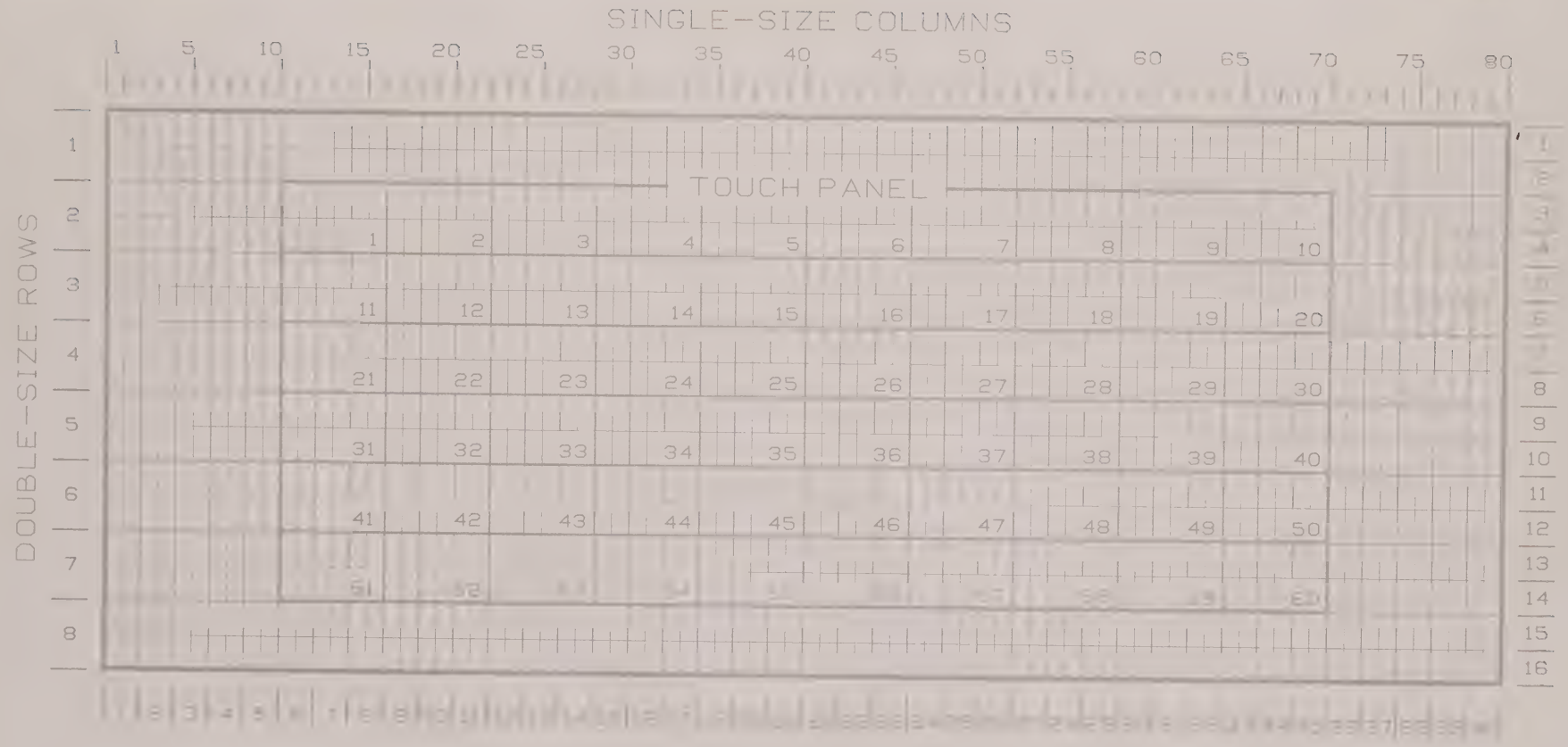
Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

FLUKE

Reorder under Fluke P/N 633547 Rev 1 6/80 (pad of 50)



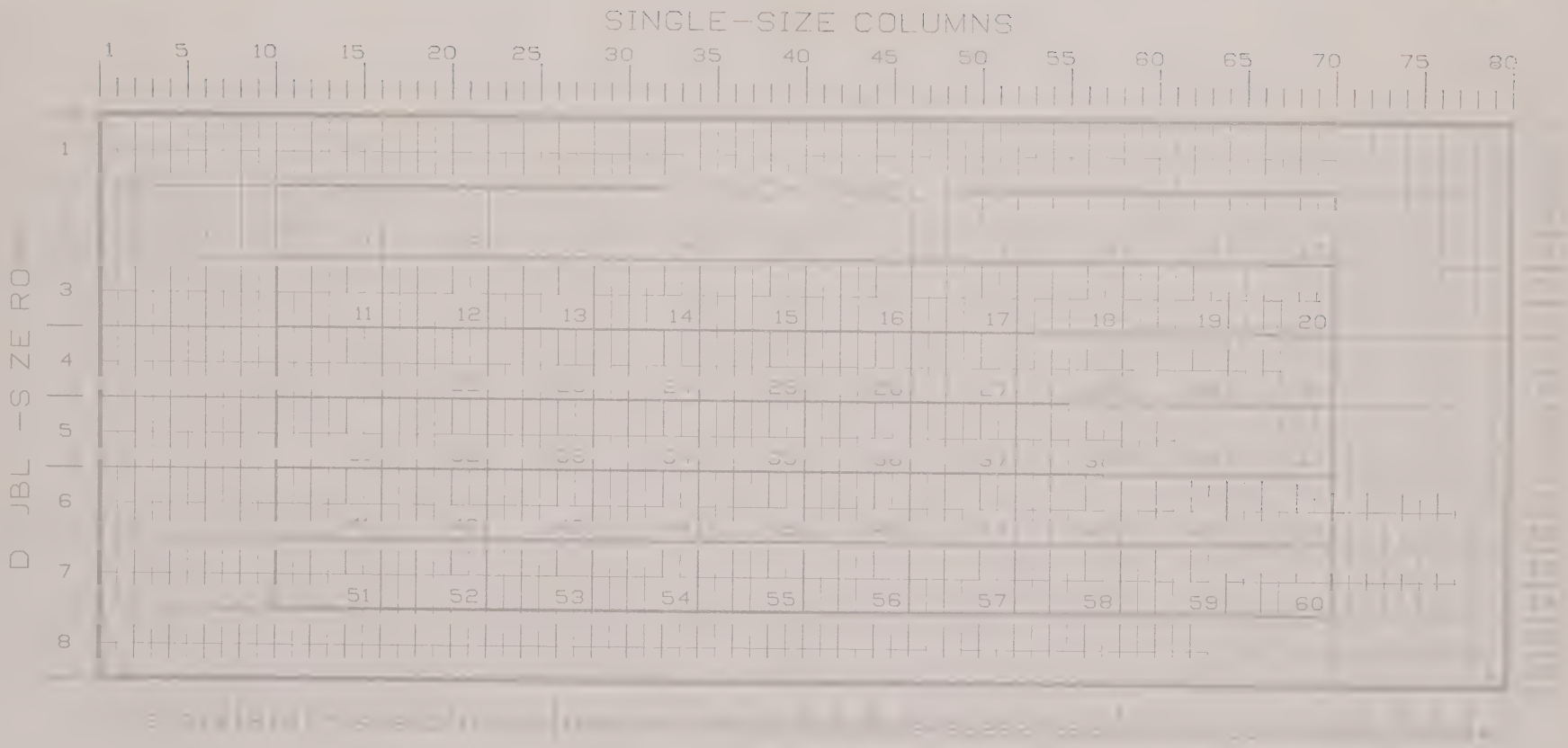
Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

FLUKE

Reorder under Fluke P/N 533547 Rev 1 6/83 (pad of 50)

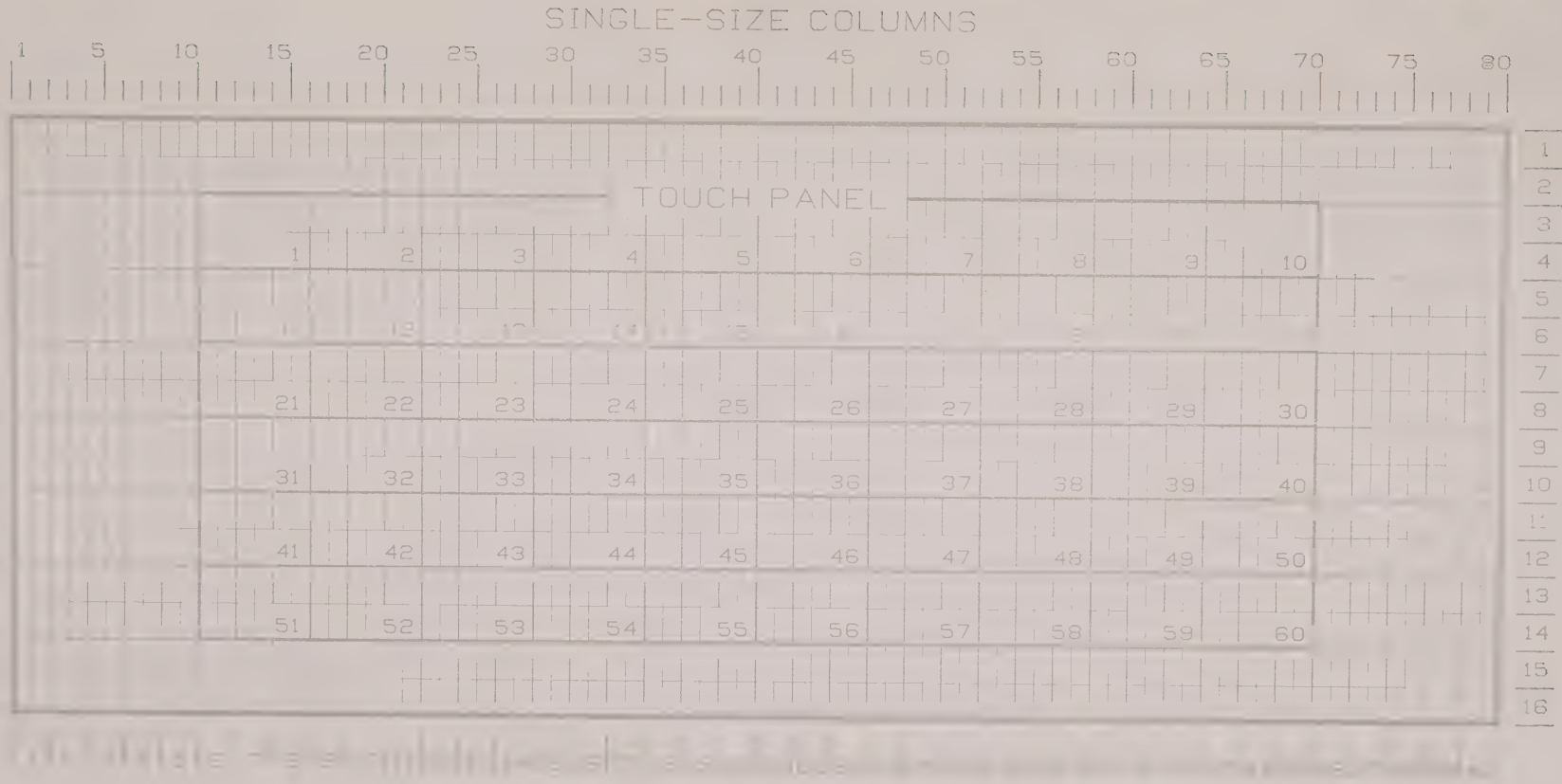


Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____

Printed under Fluke P/N 933547 Rev. 1 6/83 (part of 60)



Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

FLUKE

Reorder under Fluke P/N 533547 Rev 1 6/93 (pad of 50)

SINGLE–SIZE COLUMNS

DOUBLE–SIZE ROWS

SINGLE–SIZE ROWS

DOUBLE–SIZE COLUMNS

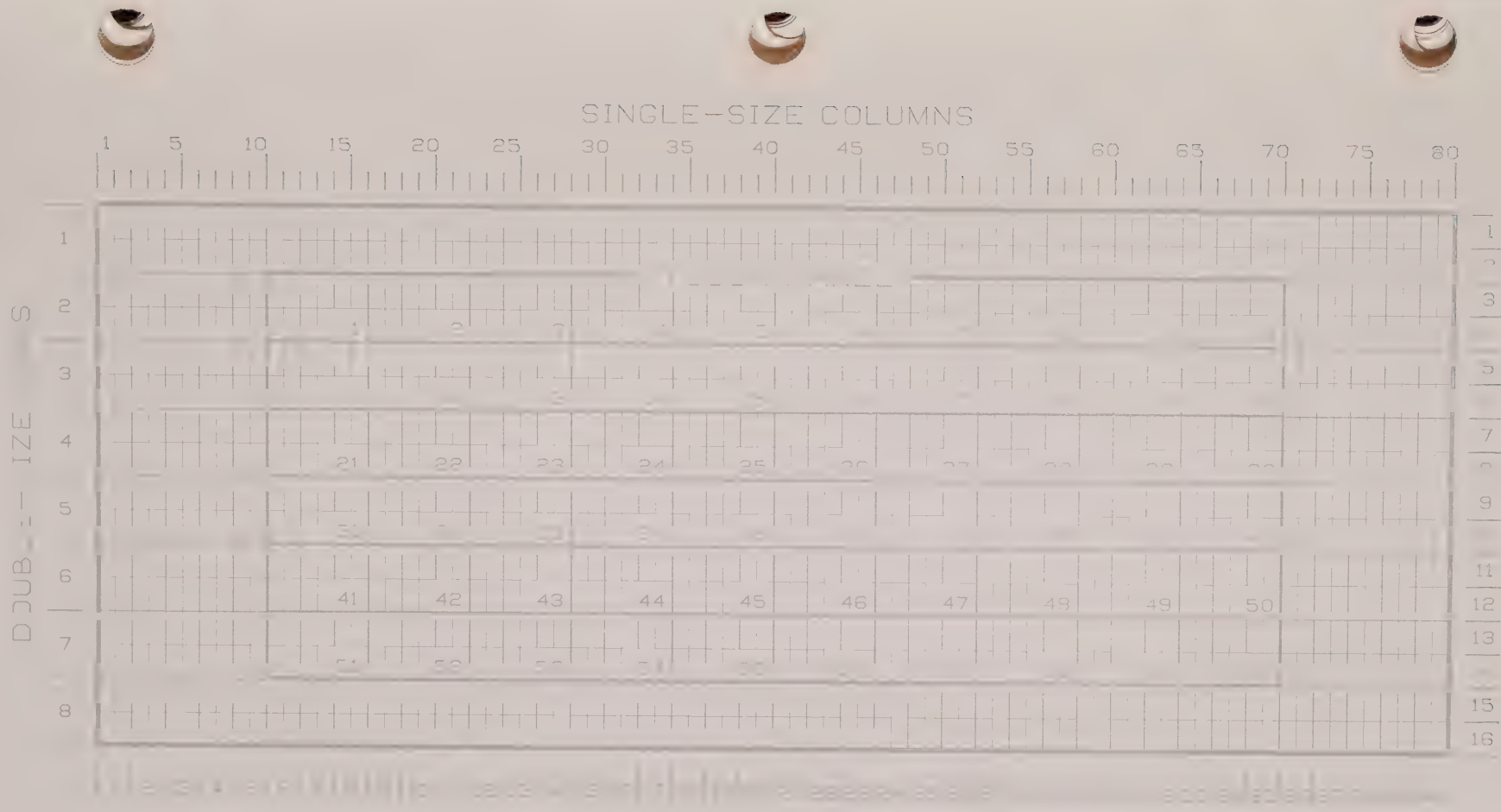
Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

FLUKE

Reorder under Fluke P/N 533547 Rev 1 6/83 (pad of 50)

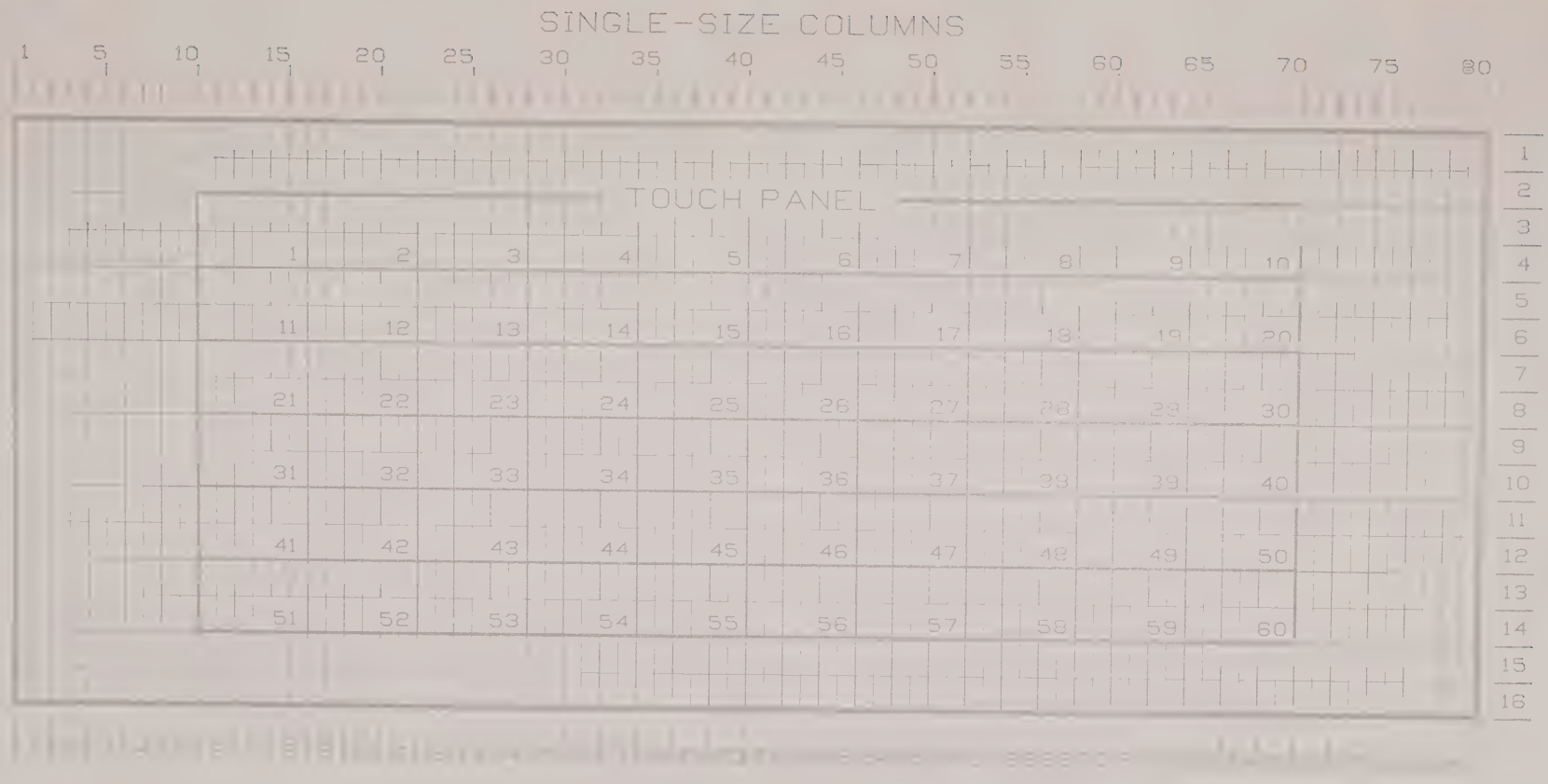


Notes:

Controller Programming Work Sheet

Program Title _____ Programmer

Reorder under Fluke P. N. 533547 Rev. 1 6/83 (pad of 50)

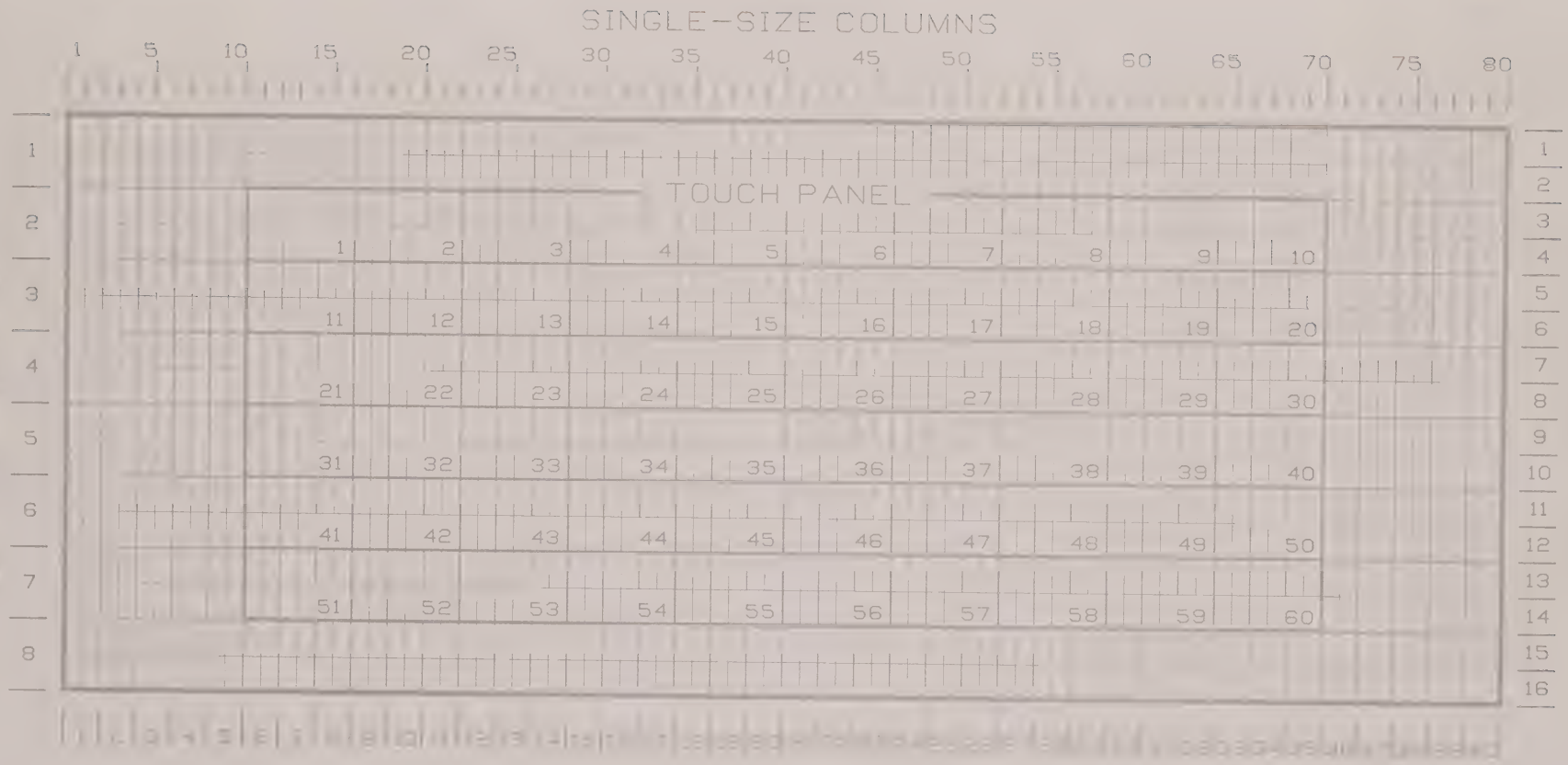


Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

FLUKE

Copyright under Fluke P/N 533541 / Rev 1 6/83 (page of 50)



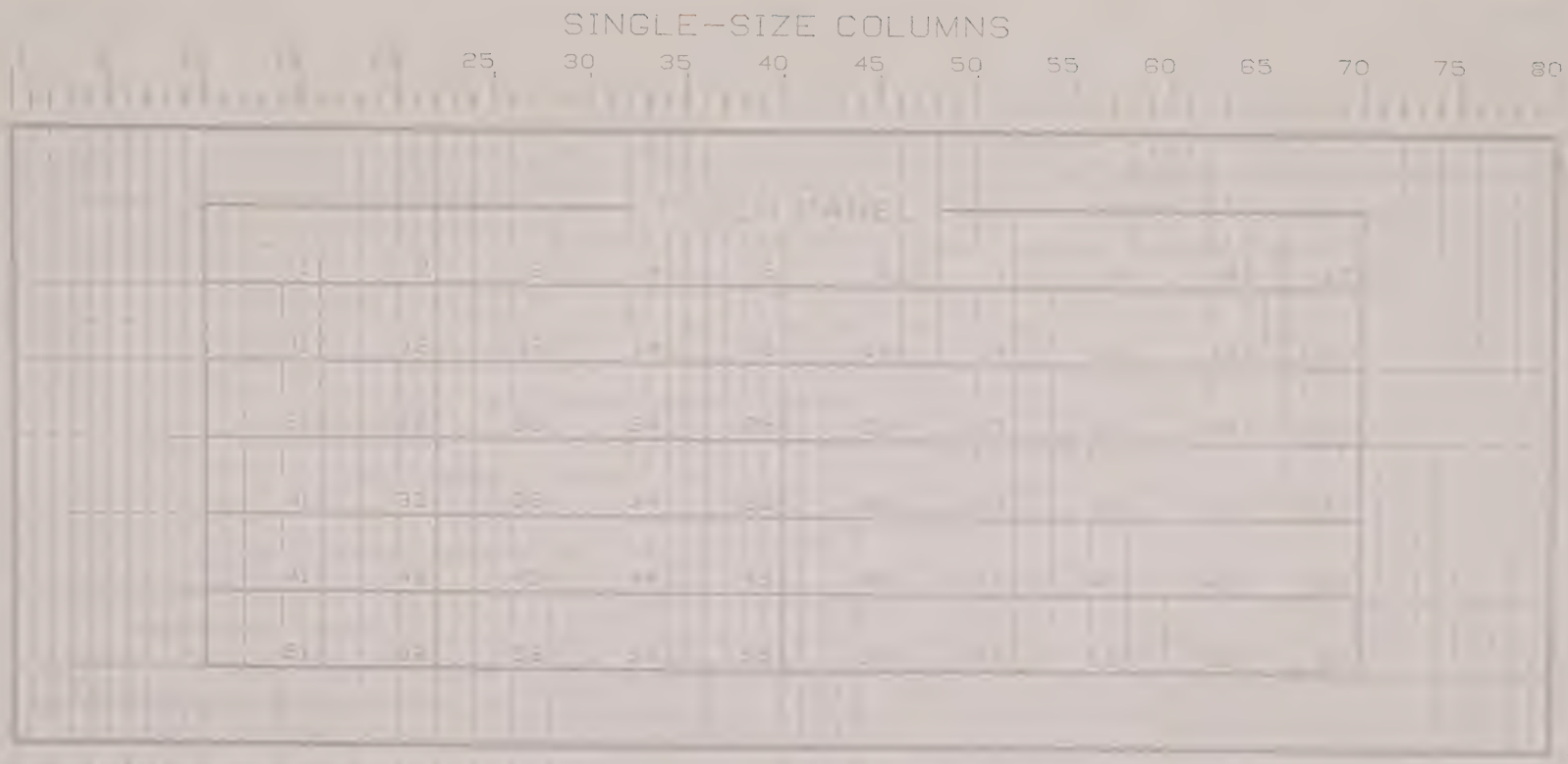
Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

FLUKE

Reorder under Fluke P/N 533547 Rev 1 6/83 (pad of 50)



Notes: _____

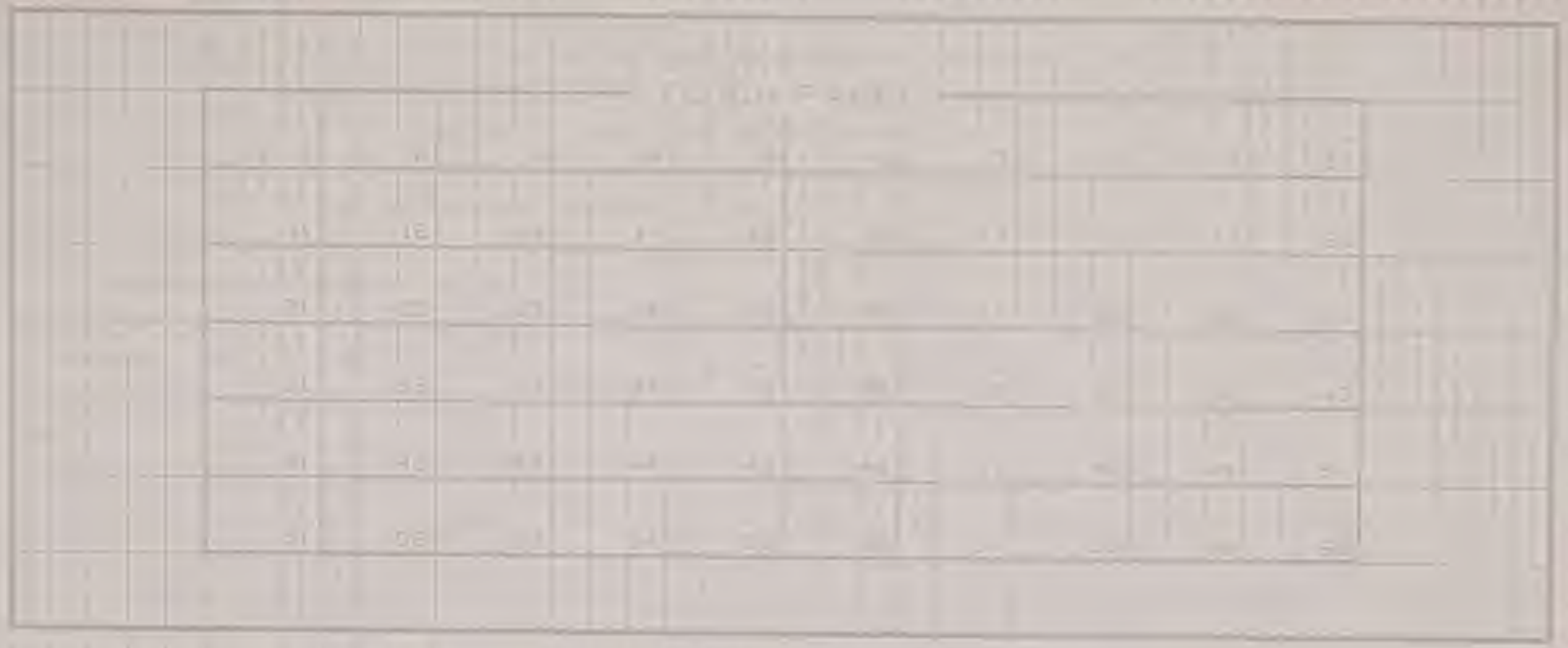
Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

FLUKE

SINGLE-SIZE COLUMNS

1 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80



SINGLE-SIZE ROWS

DOUBLE-SIZE COLUMNS

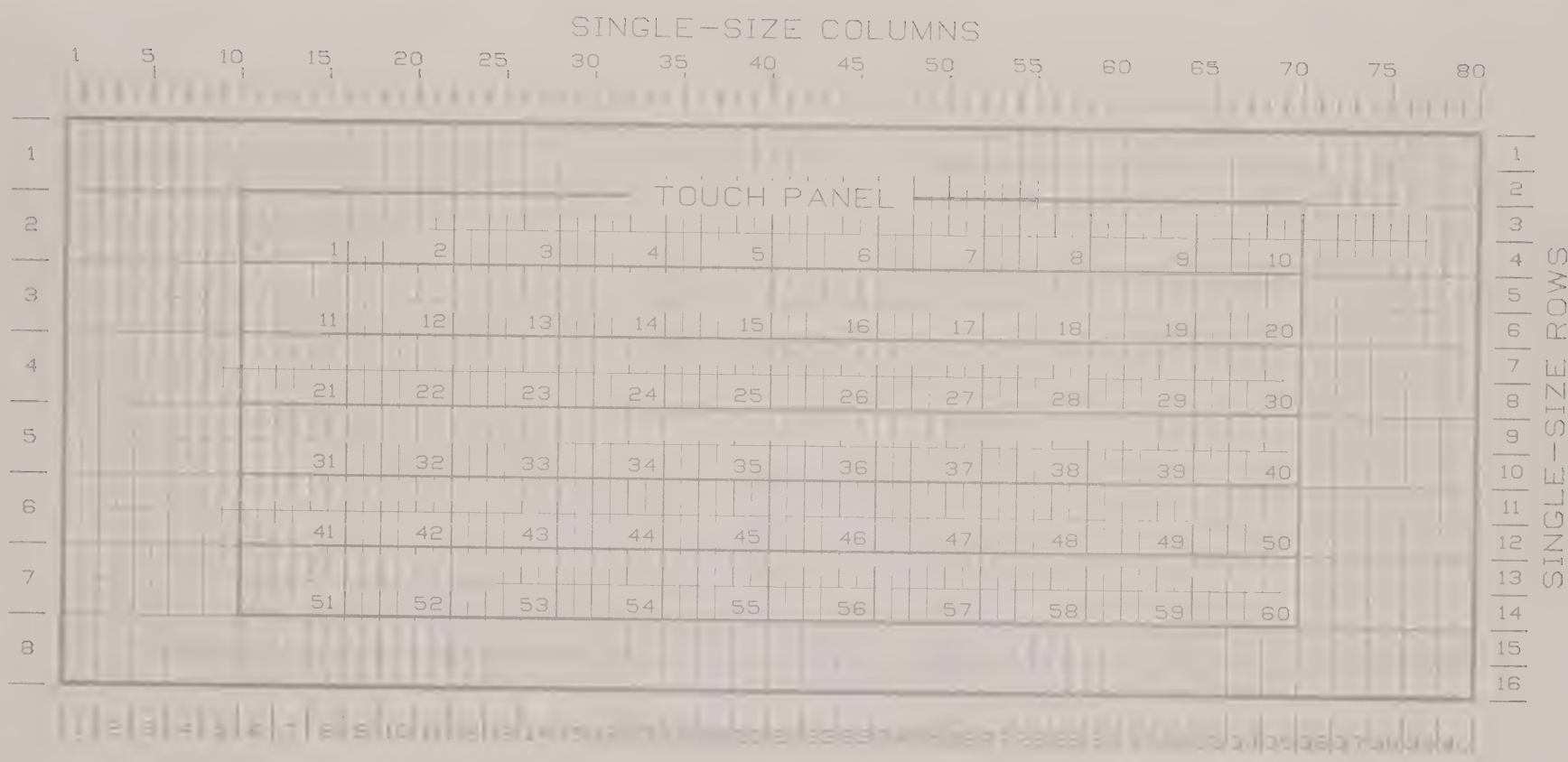
Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

FLUKE

Recorder Chart Fluke P/N 533542 Rev 1 6/'8< 'pad of 50'



Notes:

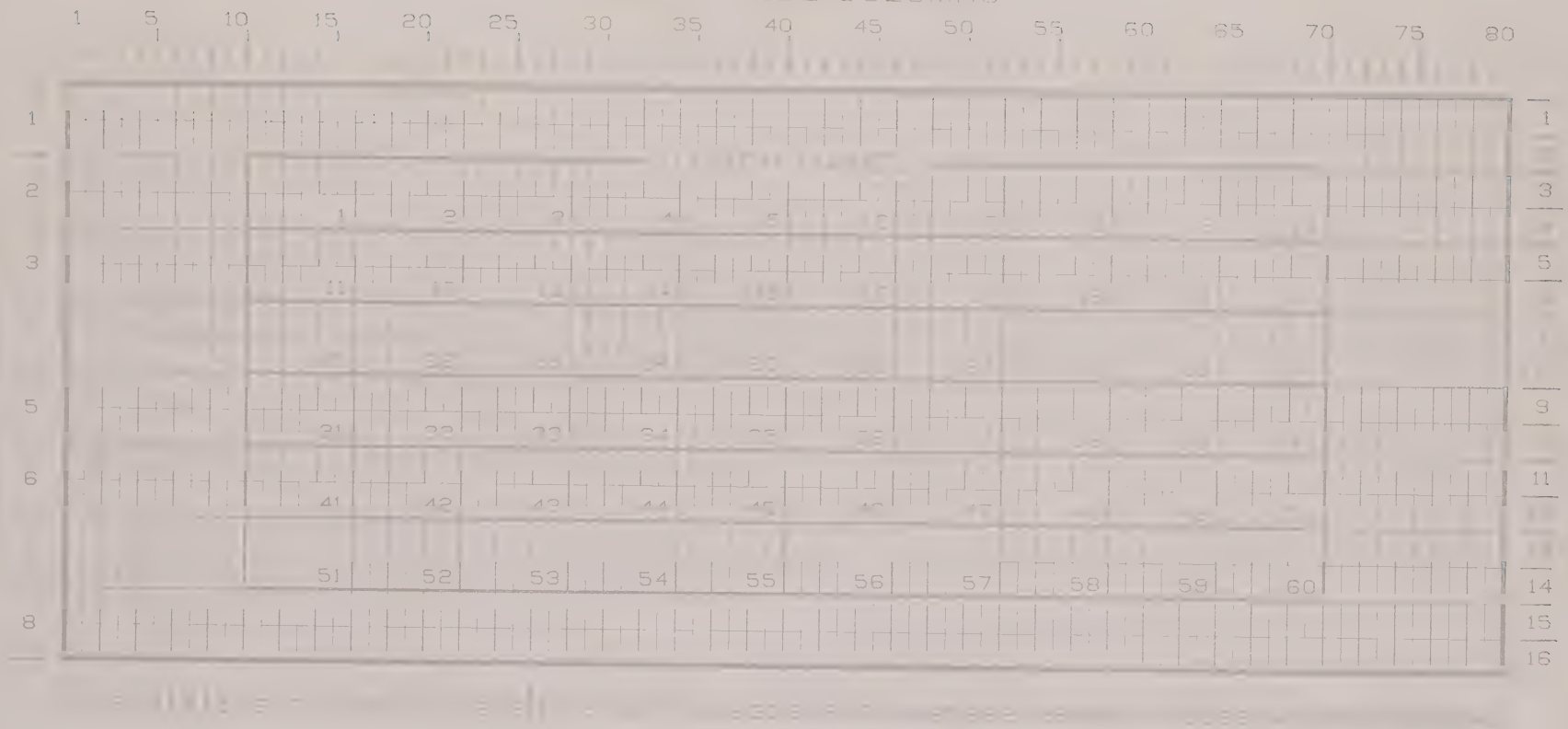
Controller Programming Work Sheet

Program Title _____ Programmer _____

FLUKE

Reorder under Fluke P/N 533547 Rev 1 6/83 (pad of 50)

SINGLE-SIZE COLUMNS



DOUBLE-SIZE COLUMNS

Notes: _____

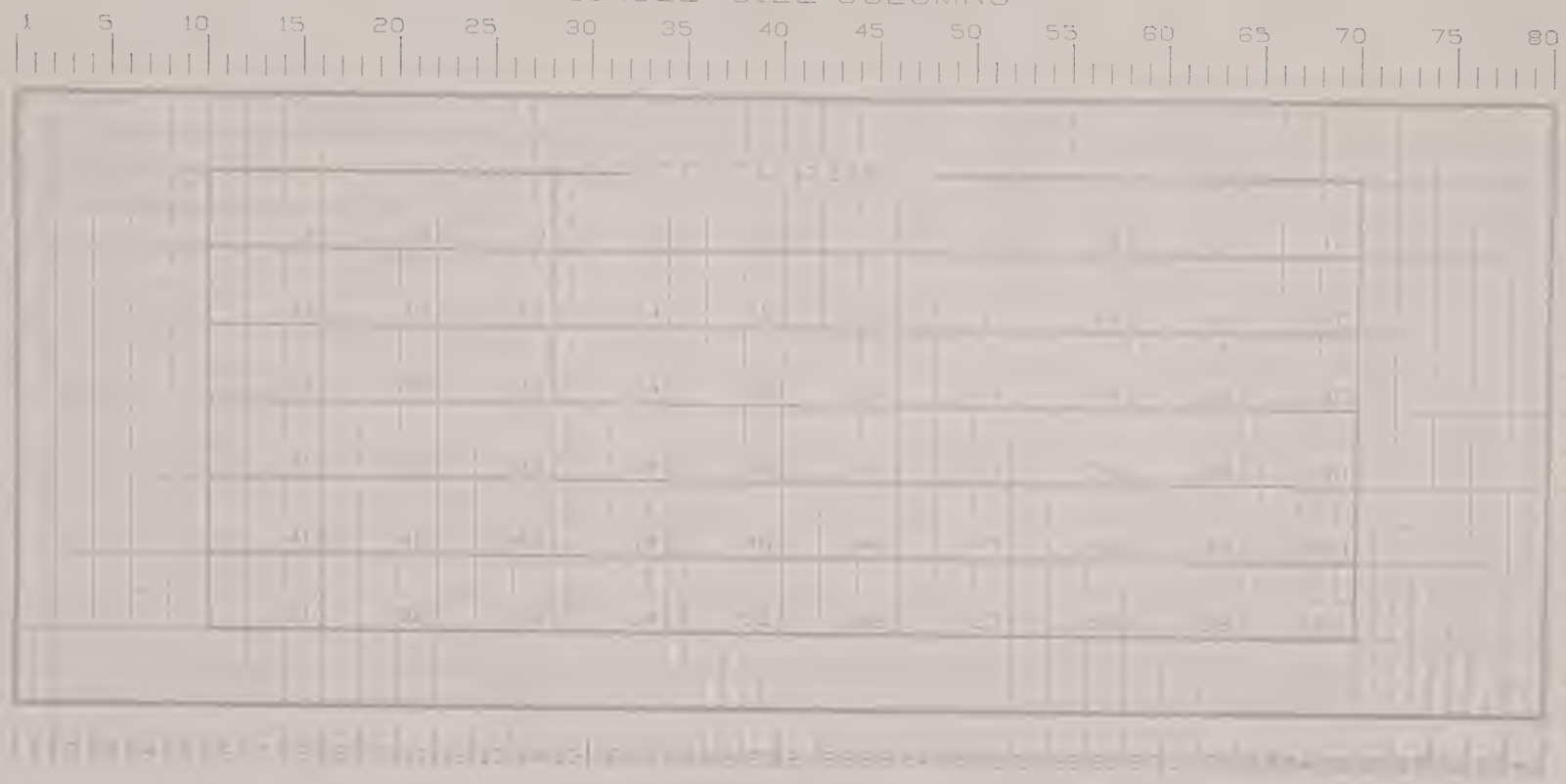
Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ____ of ____

FLUKE

Fluke Base P.N. 5335 17 Rev. 1 6/82 3 used at 50

SINGLE–SIZE COLUMNS



DOUBLE–SIZE ROWS

SINGLE–SIZE ROWS

DOUBLE–SIZE COLUMNS

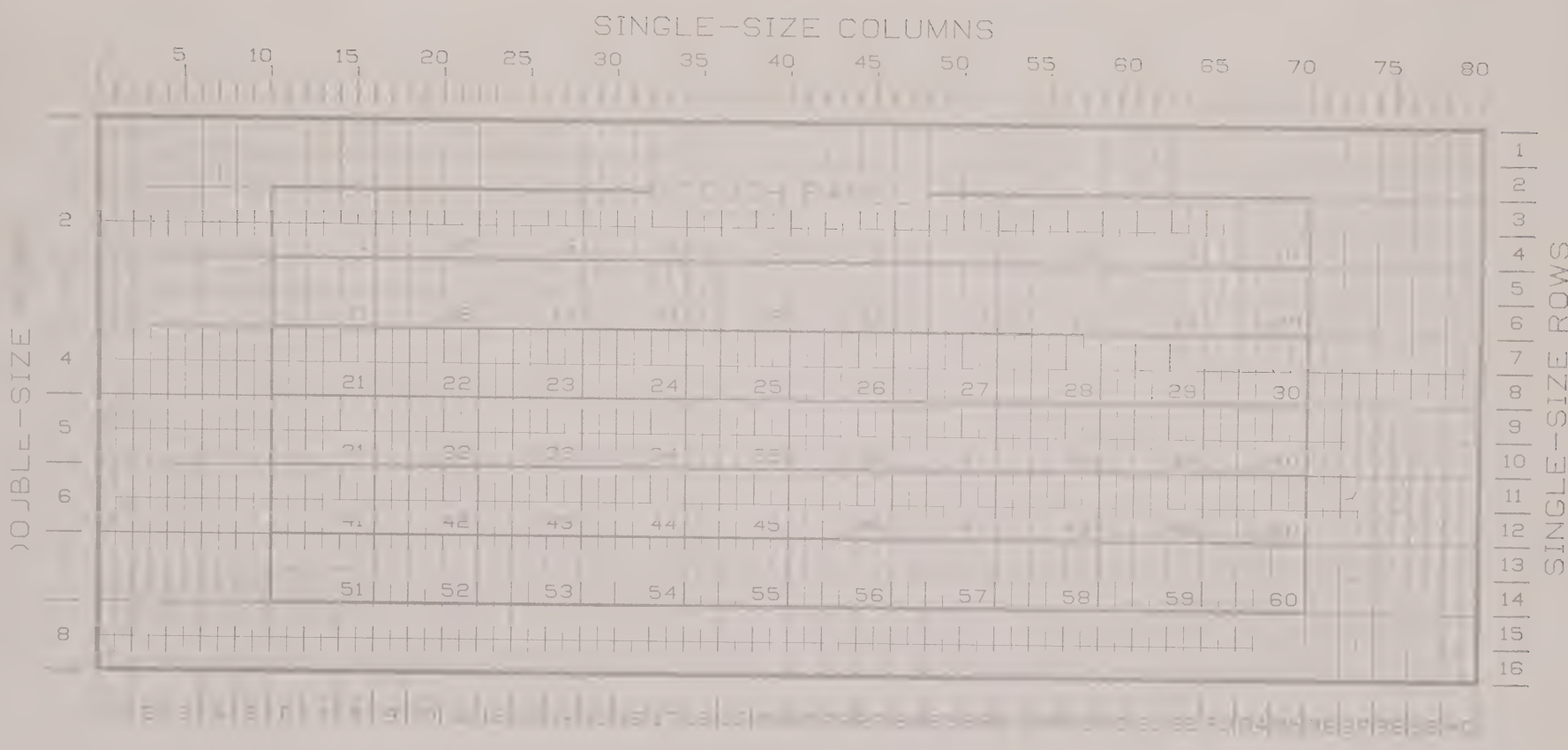
Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

FLUKE

Fluke P/N 53354; Rev. 1 6/83 (end of 50)



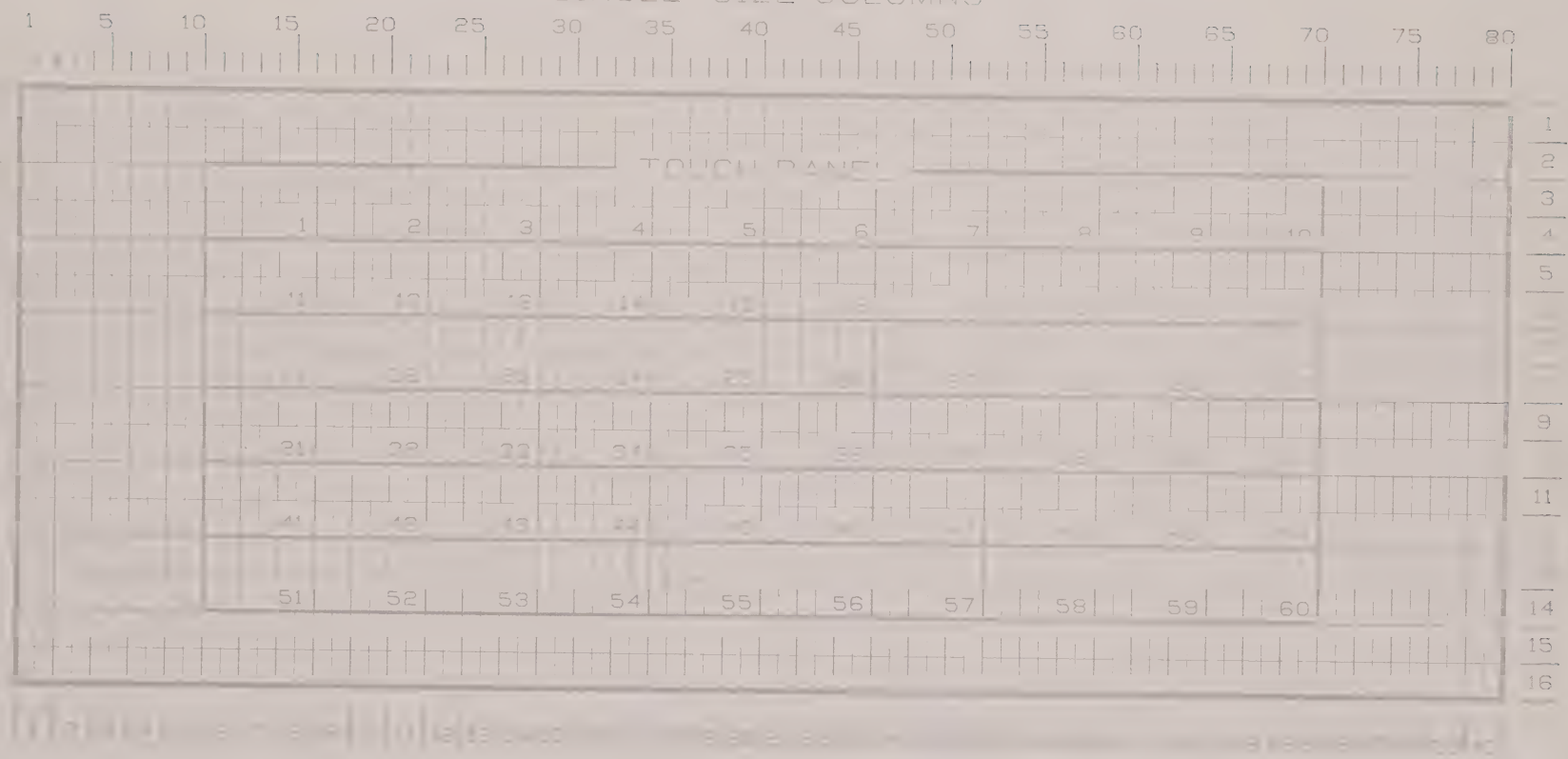
Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

FLUKE

Boarder under Fluke P/N 533547 Rev 1 6/83 (pad of 50)



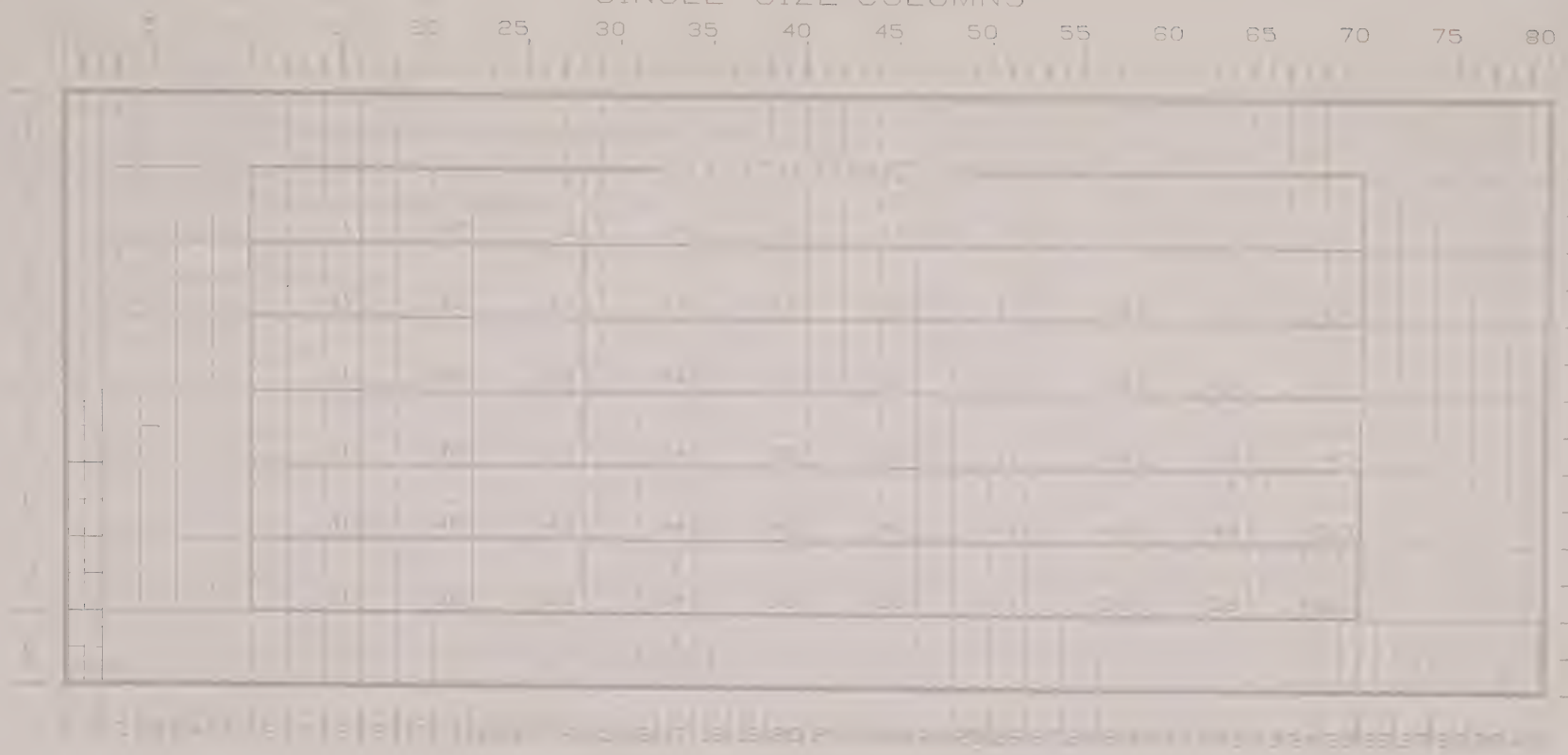
Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

Reorder paper Fluke P/N 533547 Rev 1 6/83 (pad of 50)

SINGLE-SIZE COLUMNS



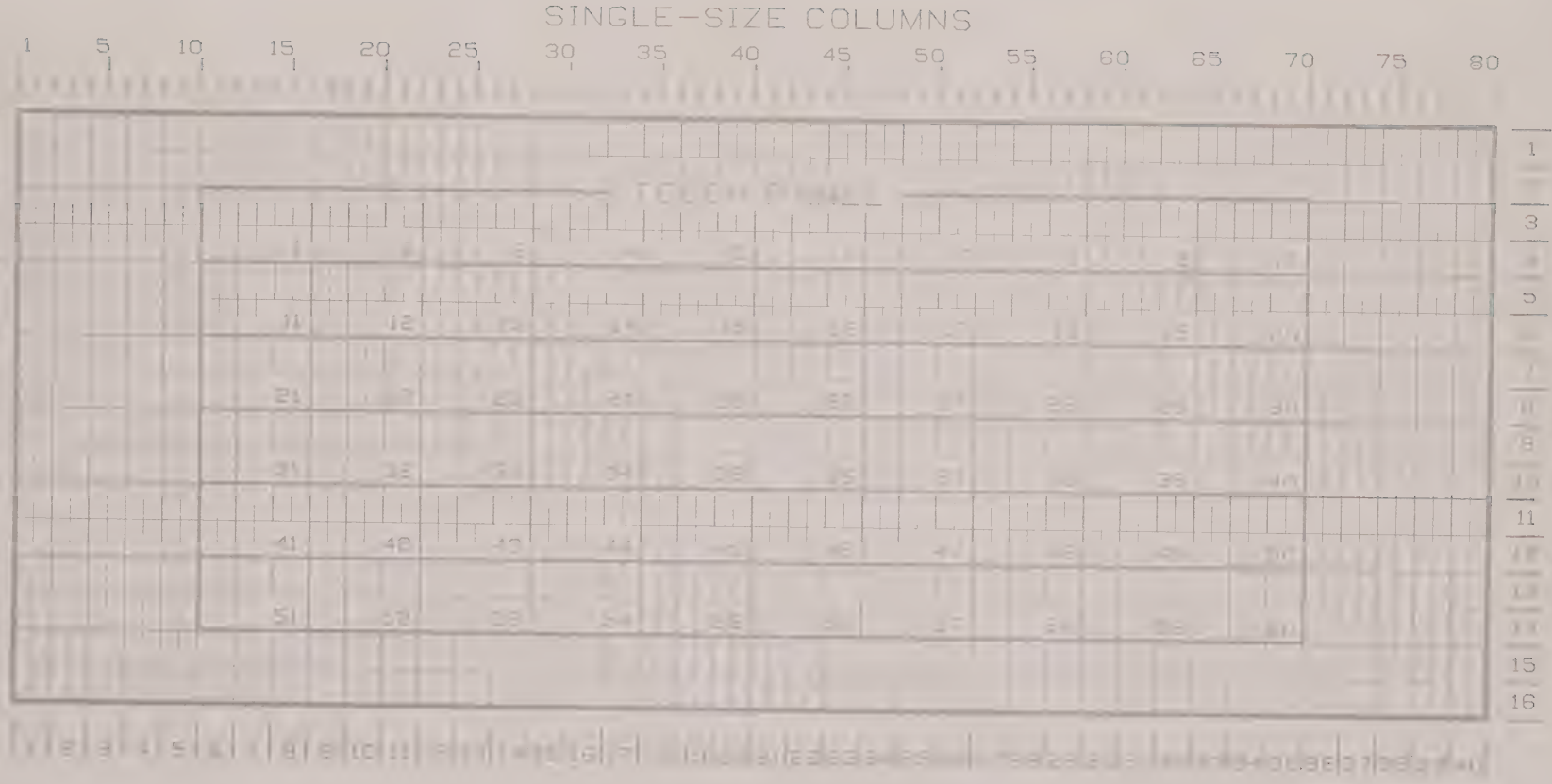
DOUBLE-SIZE COLUMNS

Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

FLUKE



Notes: _____

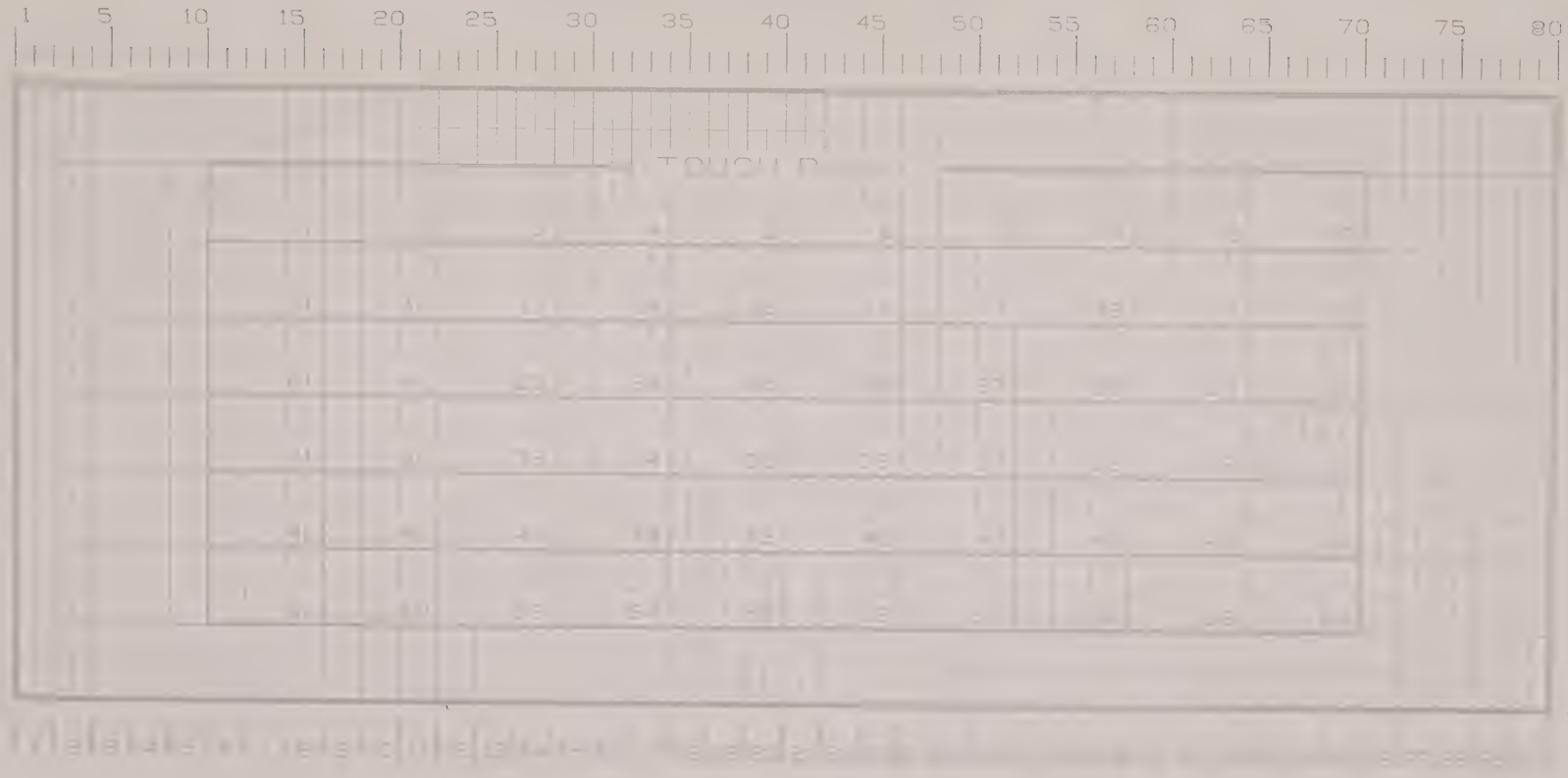
Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

FLUKE

Reorder under Fluke P/N 533547 Rev 1 6/83 (pad of 50)

SINGLE–SIZE COLUMNS



DOUBLE–SIZE ROWS

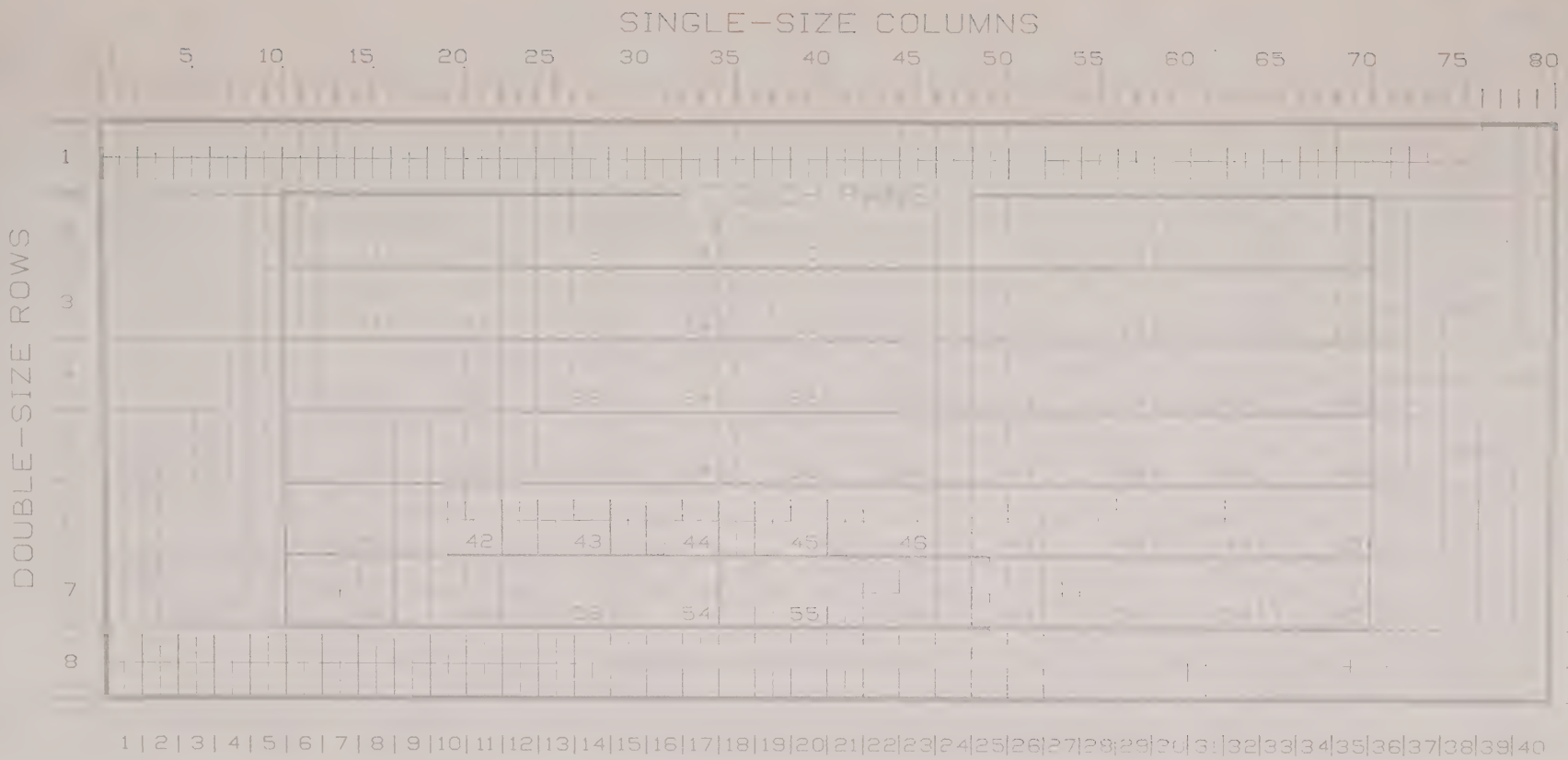
SINGLE–SIZE ROWS

DOUBLE–SIZE COLUMNS

Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

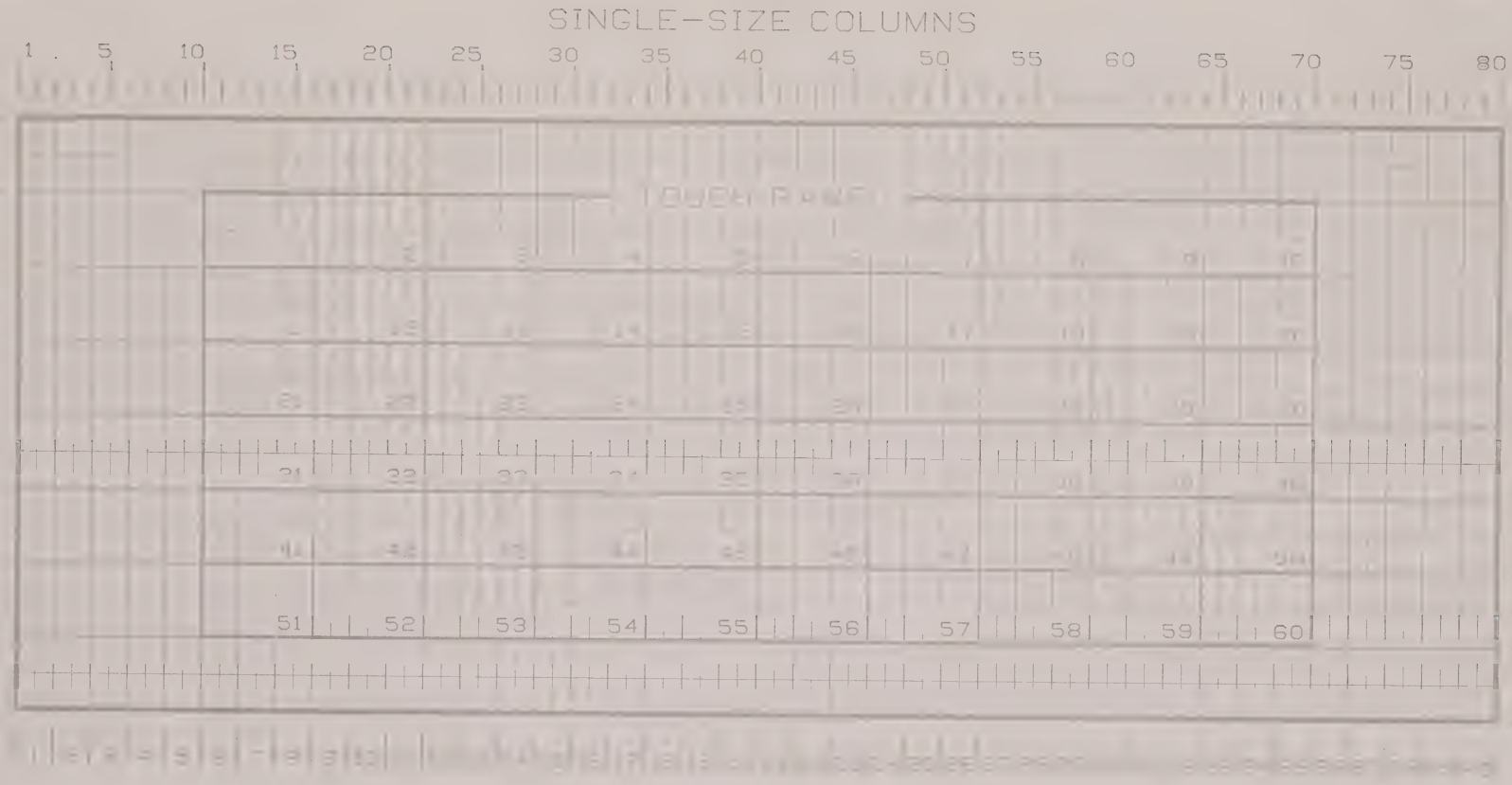


Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

FLUKE

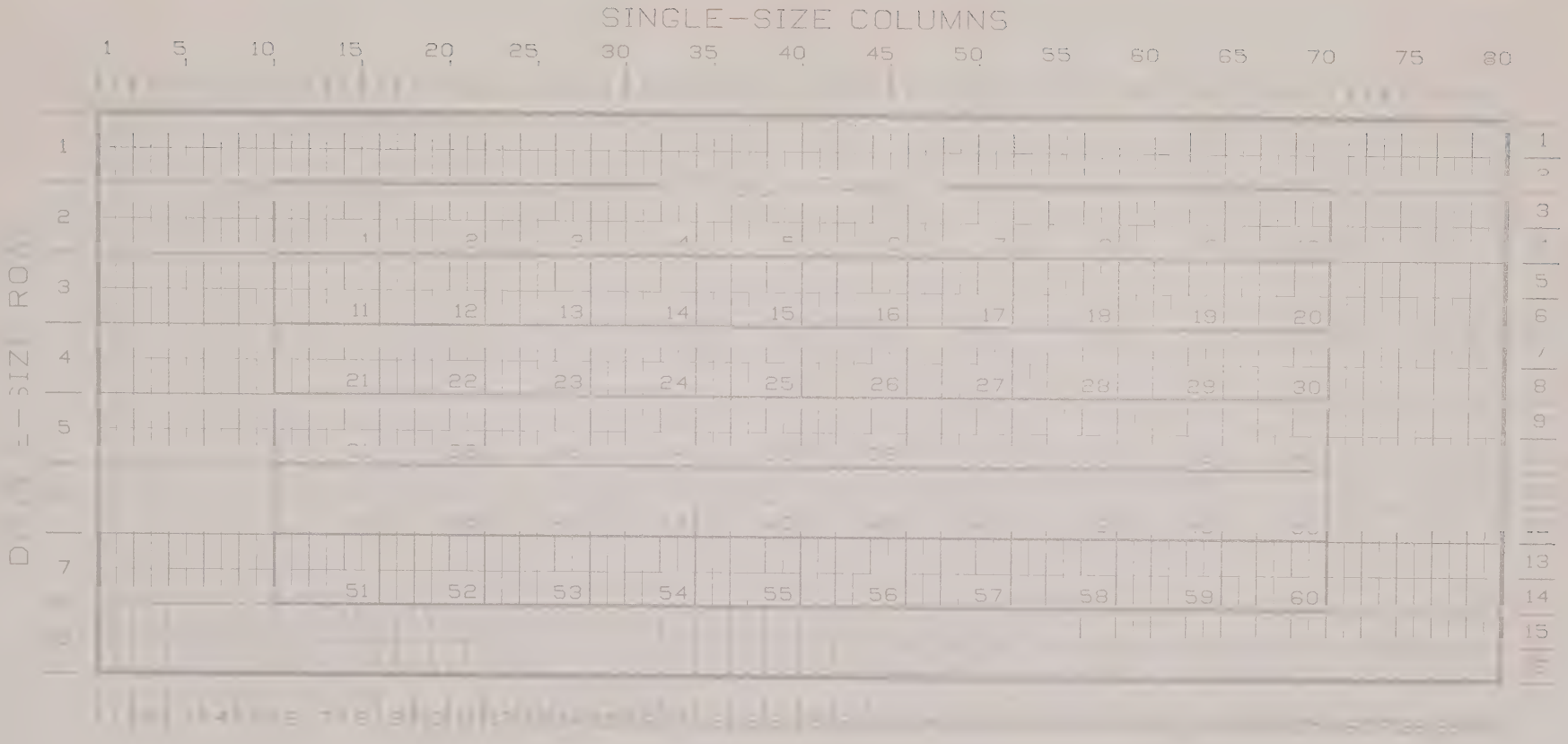


Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

FLUKE

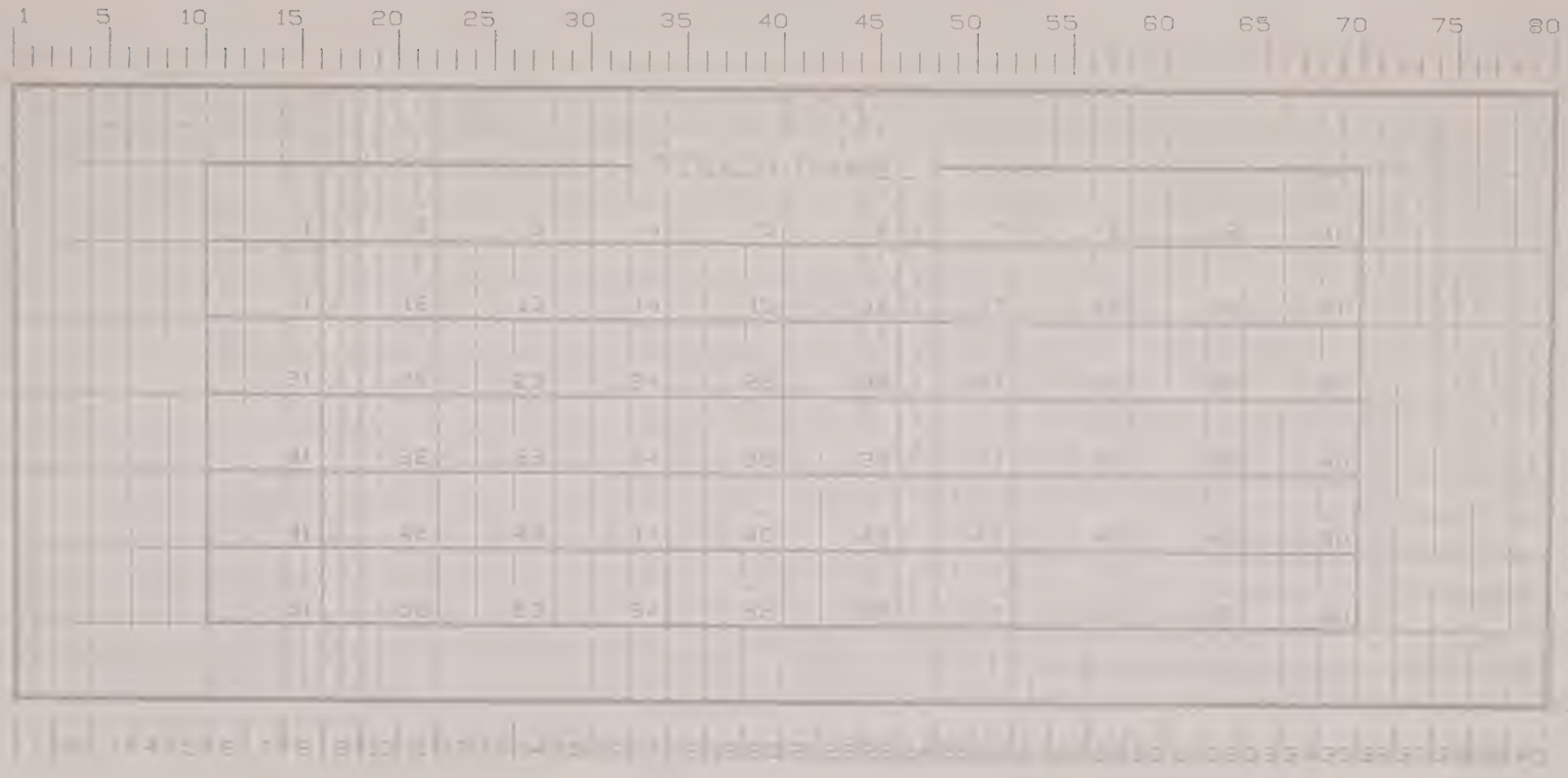


Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____

SINGLE–SIZE COLUMNS



DOUBLE–SIZE ROWS

SINGLE–SIZE ROWS

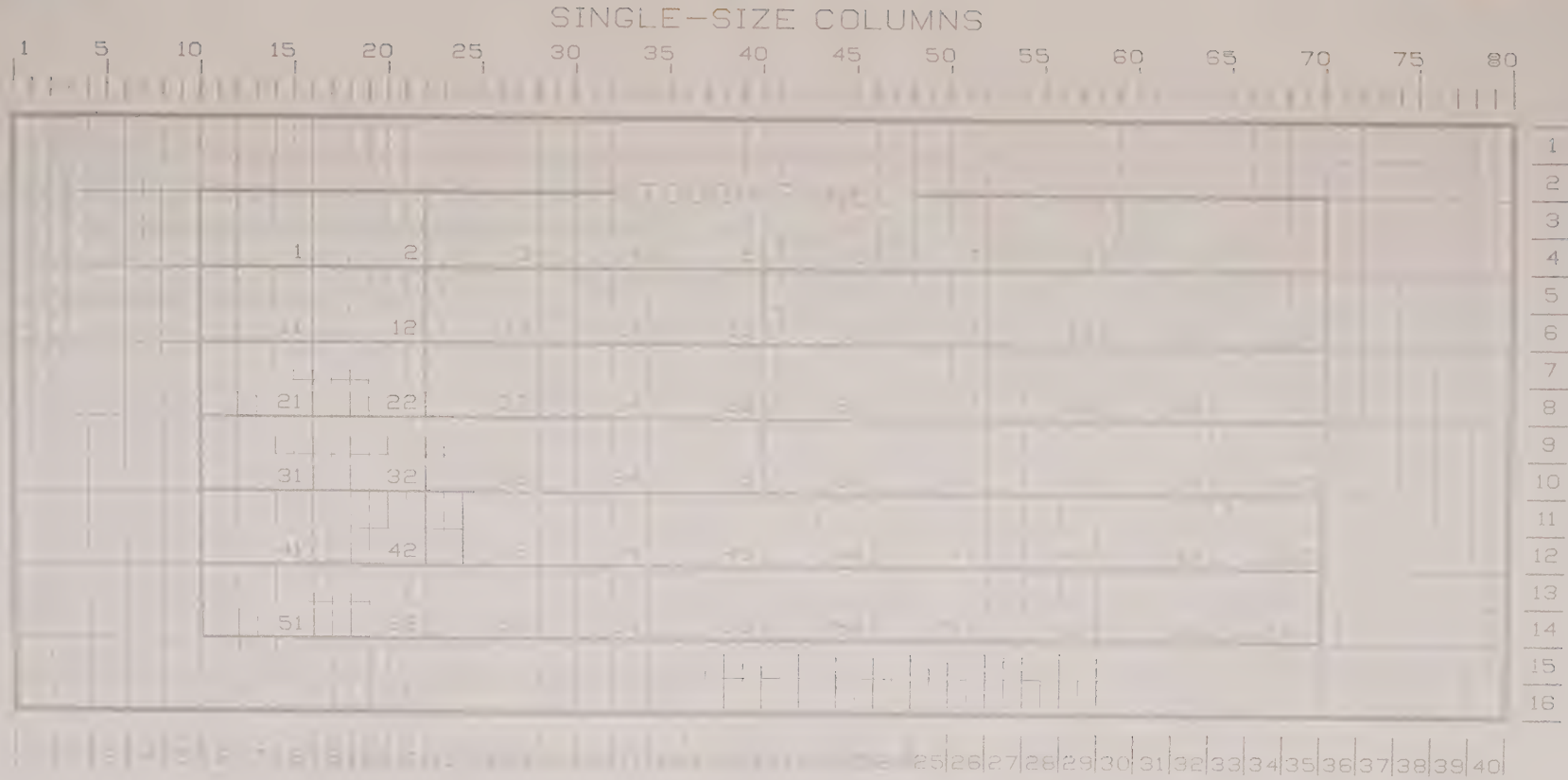
DOUBLE–SIZE COLUMNS

Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

FLUKE



Notes: _____

Controller Programming Work Sheet

Program Title _____ Programmer _____ Page ___ of ___

FLUKE

Imprimé sous Fluke P-N 53354 / Rev 1 6/93 (pad at 0)

1720A
Training Package

P/N 648287
January, 1982
©1982 John Fluke Mfg. Co., Inc., all rights reserved. Litho in U.S.A.

FLUKE®

Technical Bulletin

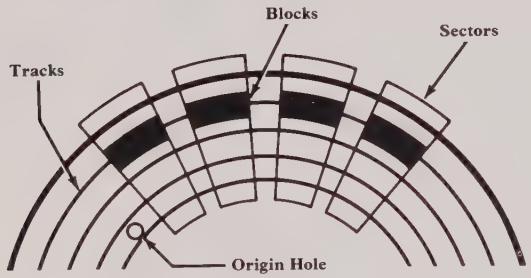
1720A Highlighted Learning Program

The Floppy Disk

The 1720A stores programs on a 5-1/4 inch floppy disk.

The disk itself is called the **MEDIA.**

The floppy disk has a smooth magnetic surface on which information is stored as sequences of magnetic pulses. The magnetic pulses are recorded along **TRACKS** on the surface of the disk.



The disk we use with the 1720A is **SINGLE SIDED** which means we can record data on only one side of the disk.

Our data is recorded using **DOUBLE DENSITY** recording techniques.

The disk is **SOFT SECTORED**, which means it has one origin hole.

It contains **35 TRACKS** and **350 BLOCKS.** Two **BLOCKS** are always used for the **DIRECTORY.** Each block stores 512 bytes (256 words) of data so the total storage capacity of the disk is approximately **175K BYTES (87K WORDS).**

Our files are stored on adjacent sectors so they are called **CONTIGUOUS FILES.**

A complete description of our program storage device is as follows:

The 1720A uses a 5-1/4 inch floppy disk. It is a **SINGLE-SIDED, DOUBLE DENSITY, SOFT-SECTORED 35-TRACK** disk. Its storage capacity is **175K BYTES (87K WORDS)**, and it has a **CONTIGIOUS FILE STRUCTURE.**

Because of the large amount of data stored on it, the floppy disk is also referred to as a **MASS STORAGE DEVICE.**

The Floppy Disk: Friend or Foe?

The floppy disk is **EASY TO USE** and allows **FAST ACCESS TO THE DATA** but if mistreated it can be your **WORST ENEMY.**

In other words, **DO NOT FOLD, SPINDLE OR MUTILATE!**

Care and Handling

- Always keep the disk in its protective cover when not in use.
- Careless handling of the disks may damage them. Avoid dropping, throwing or twisting the disks.
- Make sure to store disks vertically. Stacking may distort them and affect their contents by pressing the side covers into the disk.
- Direct sunlight may warp the disks. See that they are protected from the heat of the sun.
- Magnetic sources may distort data on the disks. Keep them away from electric motors, generators and transformers.
- Use a felt tip pen to mark disk labels. Pressure from a ball point pen or pencil may distort the data on the disk.

Loading the Disk

To load the disk, open the front disk entry latch and gently insert the disk label up, slot side first, into the disk entry latch using partial closures to seat the disk. Always grasp the disk by the cardboard cover and avoid touching the disk itself.

Note: If the door is closed when the disk is improperly seated, the disk center hole may be damaged. This can be prevented by partially lowering and raising the disk entry latch to seat the drive hub prior to total closure of the disk entry latch.

Formatting the Disk

- In order to write onto a blank disk it is necessary to go through a preliminary step called **FORMATTING.**

- During the formatting step, the floppy disk drive designates sectors and tracks using appropriate magnetic codes. This is an automatic process which the 1720A performs using the **FILE UTILITY PROGRAM (FUP).**
- The command for formatting is **/F.**
- Once the disk is formatted, it is ready to use. You can **WRITE** information onto it and **READ** it back.

John Fluke Mfg. Co., Inc.
P.O. Box C9090, Everett, WA 98206
800-426-0361 (toll free) in most of U.S.A.
206-356-5400 from AK, HI, WA
206-356-5500 from other countries

Fluke (Holland) B.V.
P.O. Box 5053, 5004 EB, Tilburg, The Netherlands
Tel. (013) 673973, TELEX 52237
Phone or write for the name of your local Fluke representative.

©1982, John Fluke Mfg. Co., Inc., all rights reserved

Printed in U.S.A.

B0076A-10U8003/SE EN

FLUKE®

Technical Data

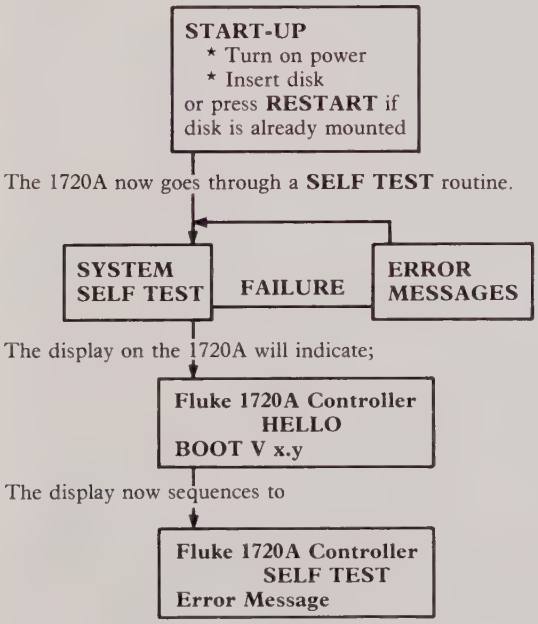
1720A Highlighted Learning Program B0077

Startup, Self Test, FDOS, Common

The 1720A controller goes through a **series** of **operations** at **start-up** to insure the **functional operation** of the **hardware** and to allow the **programmer/operator** to get into **BASIC** quickly and easily. Each operation is presented in the order in which it occurs.

Start-up follows a logical sequence which we will go through step-by-step.

The **SELF TEST** is loaded from the **ROM's (READ ONLY MEMORY)** located on the CPU board. We will follow the **START-UP** procedure step-by-step as detailed in the included flowchart.



SELF TEST performs a test on the following items:

***VIDEO BOARD**
***MEMORY BOARD**

***FLOPPY DISK INTERFACE**
***IEEE INTERFACE**

If any of the items do not test properly an **ERROR MESSAGE** will appear.

!VIDEO ERROR
!MEMORY ERROR

!FLOPPY ERROR
!IEEE MISSING OR FAULTY

If any **SELF TEST** errors are encountered, the display will also indicate

"Press any key to continue"

Pressing any key on the keyboard or Touch Sensitive Display will cause the start-up procedure to continue.

The display then sequences to

FLUKE 1720A CONTROLLER
Loading
ERROR MESSAGE

The 1720A now checks to see if a floppy disk is mounted. If so it checks to see if an operating system is present and if so it then loads **FDOS.**

If a floppy disk is not mounted or the operating system is not present on the floppy disk, the 1720A performs the same checks on the E-disk. If the operating system is found first on the E-disk, then the E-disk becomes the system device. If the operating system is found first on the floppy disk, then the floppy disk becomes the system device. If any errors are detected, one of the following error messages will appear:

?FLOPPY ERROR
?DISK NOT MOUNTED

?NO SYSTEM ON DEVICE
?ILLEGAL DIRECTORY

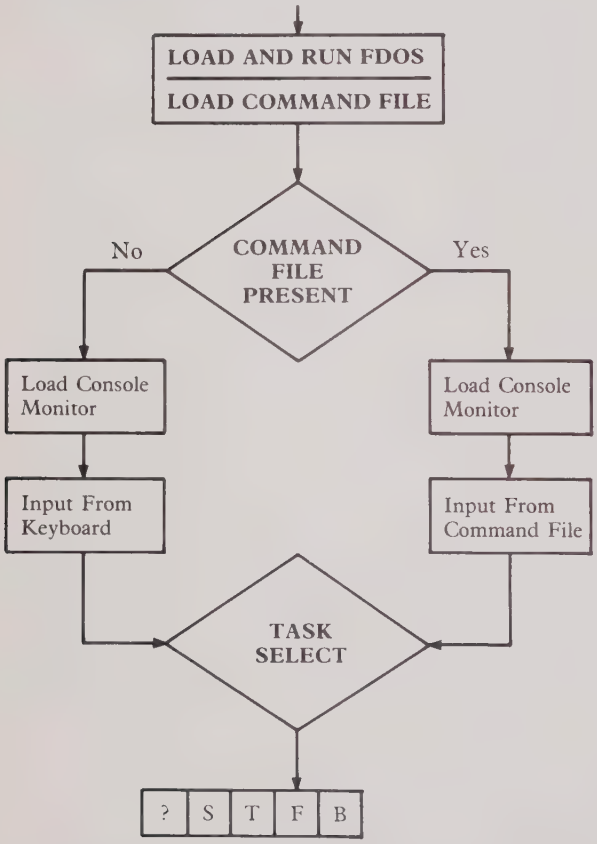
along with the message **"Press any key to continue."**

To continue after an error is encountered (such as **"?DISK NOT MOUNTED"**) and the error is corrected (mounting the disk), press any key on the keyboard or Touch Sensitive Display to continue the procedure.

FDOS IS NOW RUN (FLUKE DISK OPERATING SYSTEM)

FDOS manages disk operations and oversees communication between the disk and the Controller.

FDOS also checks for the presence of a **COMMAND FILE.**



The **COMMAND FILE** stores a set of instructions that normally would be input by the operator from the keyboard before **BASIC** is called and the **MAIN PROGRAM** is executed. It executes the preprogrammed instructions and can call up **BASIC** and run the main program. It allows, therefore, a set of start-up instructions to be executed automatically from the disk without the use of a keyboard. The **COMMAND FILE** will be covered in detail in a later lesson after keyboard execution of these instructions is learned. If no **COMMAND FILE** is present the 1720A now loads and runs **CONMON.**

CONMON stands for **CONSOLE MONITOR.** This is the program that oversees communication between the user and the 1720A. It allows selection of various utilities and programs. It asks the user to respond to the prompt character #.

The selections Are:

?displays a menu of the selections.
S SET BAUD RATE UTILITY
T SET TIME/DATE UTILITY
F FUP (FILE UTILITY PROGRAM)
B BASIC INTERPRETER

At start-up, the baud rate defaults to 4800 baud. (Internally selectable) The **SET BAUD RATE UTILITY** allows the baud rate on the two RS-232 ports (KB1, KB2) to be changed to the desired baud rate. An example of how to set the baud rate is shown below:

*The underlined portions indicate the user responses.
*(CR) stands for carriage return.

Example:

```
Console Monitor Versions x.y  11:08  26-SEP-79
# S (CR)
SET 11:08
SET Version x.y
Enter baud rate:
KB1 (J23):  2400 (CR)
KB2 (J22):  110 (CR)
```

After the KB2 baud rate is entered, the 1720A returns to **CONMON** and displays:

```
Console Monitor Version x.y  11:00  26-SEP-79
#
```

The **SET TIME/DATE UTILITY** sets the internal clock of the 1720A. Once it is set, it is supported by battery back-up even though power is removed from the 1720A. The procedure is shown below:

```
Console Monitor Version x.y  11:08  26-SEP-79
# T (CR)
TIME  11:08  26-SEP-79
Enter date:  DD-MM-YY  27-9-79 (CR)
Enter time:  HH-MM  11-14 (CR)
```

After the time is entered the 1720A returns to **CONMON** and displays the prompt #.

The **FILE UTILITY PROGRAM (FUP)** is a file transfer and management program. It is a versitile and often used program and will be covered separately in a later lesson. **FUP** allows operations such as listing the file directory, saving, merging, deleting, copying, and renaming files. It is also used to format and pack a disk. The procedure for calling **FUP** is shown below:

```
Console Monitor Version x.y  11:08  26-SEP-79
# F (CR)
FUP  11:08  26-SEP-79
File Utility Program Version x.y
*
```

To exit from **FUP** back into **CONMON**, use the procedure shown below.

Example:

```
* /X (CR)
```

or

```
* CTRL P
```

CTRL P means holding down the CTRL button and pressing the **P** button.

In order to **RUN** a **BASIC PROGRAM**, it is necessary to call up the **BASIC INTERPRETER**. It is called up from **CONMON** and is loaded into **MAIN MEMORY**. We can then **LOAD** the **USER'S BASIC PROGRAM** into **MAIN MEMORY** and **RUN** it. This will be covered in detail in a later lesson. The procedure for calling **BASIC** is shown below:

Example:

```
Console Monitor Version x.y  11:08  26-SEP-79
# B (CR)
BASIC  11:08
BASIC Version x.y
Ready
```

There are two ways to run a **BASIC** program. One way is to **LOAD** the program and then **RUN** it.

Example:

```
Ready
OLD "METER" (CR)
Ready
RUN (CR)
```

The other way is to **LOAD** and **RUN** the program with one command.

Example:

```
Ready
Run "METER" (CR)
```

In order to **HALT** a program that is running, the operator can enter the following:

```
CTRL C
Ready
```

A program can also be halted using the **ABORT** button on the front panel. This button **HALTS** the program and issues a **DEVICE CLEAR** over the **IEEE-488 BUS.**

Another way to **HALT** a program is to use the **CONTROL P** function. This function **EXITS** from **BASIC** and returns to **CONMON** (unless the program had been **SAVED** on the **DISK** or **E-DISK**, it would be lost).

Example:

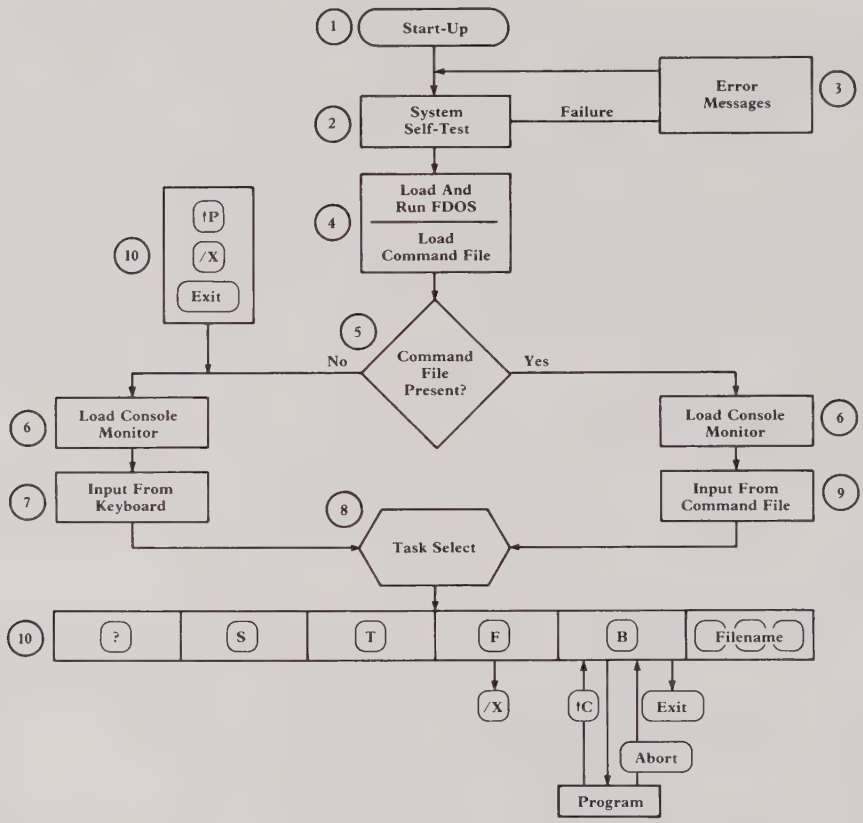
```
CTRL P
Console Monitor Version x.y  11:08  26-SEP-79
#
```

Another way to **HALT** the program and **EXIT** from **BASIC** is shown below.

```
CRTL C
Ready
EXIT (CR) or CTRL T
Console Monitor Version x.y  11:08  26-SEP-79
#
```

Using **CTRL T** instead of a **(CR)** will also cause a **(CR)** to be returned but in addition will **CLEAR** the **SCREEN** and **HOME** the **CURSOR.**

START-UP, SELF TEST, FDOS, CONMON, FLOW CHART



TASK	DESCRIPTION	EXITS
?	Displays a 'menu' of the various tasks performable.	None Required
S	Set Baudrate Utility	None Required
T	Set Time/Date Utility	None Required
F	File Utility Program	CTRL P, /X
B	BASIC Interpreter	CTRL P, EXIT
Filename	Uses any of the Assembly Language Programs	As Programmed

John Fluke Mfg. Co., Inc.
P.O. Box 43210, Mountlake Terrace, WA 98043
800-426-0361 (toll free) in most of U.S.A.
206-774-2481 from AK, HI, WA and Canada
206-774-2398 from other countries

Fluke (Holland) B.V.
P.O. Box 5053, 5004 EB, Tilburg, The Netherlands
Tel. (013) 673973, TELEX 52237
Phone or write for the name of your local Fluke representative.

Printed in U.S.A.

B0077B-10U8203/SE EN

FLUKE®

Technical Data

17XXA Highlighted Learning Program B0078

FUP Fundamentals

Overview

FUP.CIL is a **file transfer** and **file management** program. This program is stored on the disk in **binary** and is called up from **CONMON.** The "**.CIL**" extension stands for **CORE IMAGE LOAD** and indicates that **FUP** is not a **BASIC** program. **FUP** is a versatile and often used program which provides the capability of performing **utility operations,** on **file structured devices,** which are not possible in the **BASIC** environment. The full list of options which can be performed by **FUP** are listed below:

COMMAND OPTION

1.	**?**	Displays a list of the FUP options
2.	**/A**	Assign the default system device (SYO:)
3.	**/B**	Binary file transfer
4.	**/D**	Delete a file
5.	**/E**	Extended directory listing
6.	**/F**	Format a file-structured device (floppy disk or E-Disk)
7.	**/L**	List the directory
8.	**/M**	Merge ASCII files
9.	**/R**	Rename a file
10.	**/S**	Search for bad blocks
11.	**/W**	Whole or partial copy
12.	**/X**	Exit from FUP
13.	**/Z**	Zero a file directory
14.	**/P**	Pack a file-structured device
15.	**/T**	Turn off error checking

In this lesson we will cover some of the **syntax** and **fundamental considerations** which are necessary in order to implement the **FUP COMMAND OPTIONS.**

FILENAMES and **EXTENSIONS** - Information such as **programs, data** or **text** which is stored on the disk (or E-Disk) is **accessed** via a **file name** with its associated extension.

NAME.EXTENSION

The **FILENAME** consists of from 1 to 6 letters, numbers, spaces or $ signs. The **extension** consists of from 1 to 3 letters, numbers, spaces, or $ signs.

Examples: "b" is used to show the space character

TEMP.BAS	FDOS.SYS	8502A.BAS
RESULT.DAT	FUP.CIL	8502A.BAL
RESULT.BIN	8502A.488	RESULT.b

There are six programs in the current operating system and an optional **COMMAND FILE.** These programs are:

FDOS.SYS	FLUKE DISK OPERATING SYSTEM
BASIC.CIL	BASIC IMMEDIATE & EDITING MODES
FUP.CIL	FILE UTILITY PROGRAM
CONMON.SYS	CONSOLE MONITOR
TIME.CIL	CLOCK & CALENDAR SETTING ROUTINE
SET.CIL	SET RS-232 BAUD RATE
COMMND.SYS	COMMAND FILE

The above **FILENAMES** and **EXTENSIONS** relate to the program stored under that **FILENAME**. The extension "**SYS**" is an abbreviation for **SYSTEM**; "CIL" is an abbreviation for **CORE IMAGE LOAD.** The system has been programmed to look for and recognize these specific **FILE NAMES** with the specific extension.

Note: *The system has not been programmed to prevent the programmer from using any of the above names with its extension to store other programs or data, and in so doing, delete the original system program!*

Every **FILENAME** gets an **extension** when it is used to **store** a program or data. The sytem assigns a **default extension** if the programmer does not use an extension.

Examples of **extensions and their origin:**

.BAS	this extension is assigned by default to any **FILENAME** which is used with a **SAVE** command in the **BASIC MODE** to store a **BASIC** program as **ASCII data. FUP** will assume a **.BAS** extension for any **FILENAME** entered without an extension.
.b	a ".space" extension is used as a default extension when a **FILENAME** is used with an **OPEN statement to store data, when the** program omits the extension.

.BIN	used by some programmers to indicate **binary data.** This is not a default extension. It has no special meaning for the system. Assigning an extension to a **BINARY** file is a good way to indicate it is not an **ASCII** file. Manipulation of **BINARY** files in **FUP** requires the **/B COMMAND**; therefore, the programmer needs to be able to recognize **BINARY** files.
.488	can be assigned to either a program or data file to indicate to the user its use with the IEEE-488 BUS. This is not a default extension; it has no special meaning to the system. "**.488**" may be used in place of "**.BAS**" and the system will still recognize the program as a **BASIC** program stored as **ASCII data**. "**.488**" may not be used in place of "**.BAL**". "**.DMO**" for demo, "**.CAL**" for calibration, "**.TMP**" for temporary, and "**.TST**" for TEST are all examples of how a group of programs could be associated by a common extension. These extensions are treated by the system the same as "**.488**".

Although **FILENAME** and **EXTENSION** are treated separately in the above discussion, the full file name of a program or data is really the **FILENAME** and the **.EXTENSION.**

Note: *In* ***FUP*** *only* ***ASCII*** *files may be listed out to a printer. If a file is stored on the disk in both the* ***.BAS*** *and* ***.BAL*** *versions (for example:* ***METER.BAS*** *and* ***METER.BAL****), and no extension is specified when using BASIC as in* ***RUN*** *"****METER****", the* ***.BAL*** *file is used. If no extension is specified when using FUP, however, the* ***.BAS*** *file is used.*

Devices the 1720A can transfer data to or from any of **six I/O DEVICES** which are designated by a **two-letter** specification plus a **single number** indicating the unit number. The designations are listed below:

KB0:	Console device (keyboard/display)
KB1:	(J23) RS-232 port
KB2:	(J22) RS-232 port
MF0:	Floppy Drive
ED0:	Electronic Disk
MM0:	Main Memory (used for copying files)

To make operation easier, there is one other device designation **SY0:**, the **SYSTEM DEVICE**. Either the floppy drive (**MF0:**) or the Electronic Disk (**ED0:**) can be assigned as **SY0:**. The 1720A loads and runs the operating system from it.

The **SYSTEM DEVICE** can be assigned from **FUP** using the /A command as shown below. The **FUP** prompt character is an *.

MF0:/A	assigns MF0: as SY0:
ED0:/A	assigns ED0: as SY0:

SYNTAX of FUP COMMANDS

Now that we know what **FILENAMES, EXTENSIONS** and **DEVICES** are, we will show how these are used to perform operations in FUP.

The general FUP command is as follows:

The format for either the output or input file is:
[<DEVICE>] : <FILENAME> [<EXTENSION>]
<OUTPUT FILE> [, <output file>] =
<INPUT FILE> [, <input file>] . <OPTION>

While multiple output files and input files may be specified (the designations in brackets [ ] are optional), we will work with the terms not enclosed in brackets as these are the minimum necessary components of a command line. This reduces the command line to:

<OUTPUT FILE>=<INPUT FILE>/<OPTION>

The **REFERENCE POINT** for all **FUP** operations is the = sign

OUTPUT FILE = INPUT FILE

Direction of Data Transfer

DATA is always transferred from the **RIGHT (INPUT** or **source FILE)** side of the = **sign** to the **LEFT (OUTPUT** or **destination FILE)** side

The default states are:

<DEVICE> = SY0
<EXTENSION> = .BAS

When **NON-FILE STRUCTURED DEVICES** like **RS-232 PORTS** and the **CONSOLE DEVICE (KB0:)** are used, a **FILENAME** and **EXTENSION** are not required. For example, **KB1:** is a valid description of an **OUTPUT FILE.**

© 1980, John Fluke Mfg. Co., Inc.,all rights reserved

We will now go through the use of the **FUP** commands. This command (?) displays the contents of the file **FUP.HLP** which summarizes the proper use and options of **FUP.** The command sequence is shown below:

/A	Assigns System Device	ED0:/A or MF0:/A
/B	Binary File Transfer	MF0:FUP.CIL=MM0:FUP.CIL/B
/D	Delete a File	TEST.TMP/D or ED0:TEST.1, TEST.2,TEST.3/D
/E	Extended Directory Listing	MF0:/E or ED0:/E or /E
/F	Format a Disk or the E-Disk	MF0:/F or ED0:/F
/L	List a Directory	MF0:/L or KB1: ED0:/L
/M	Merge ASCII Files	TEST.NEW=TEST.1, TEST.2, TEST.3/M
/P	Pack a Device	MF0:/P or ED0:/P or /P
/R	Rename a file	TEST.1=TEST.OLD/R
/S	Search for Bad Blocks	MF0:/S or /S
/T	Turn off error checking	EDO:TEST.BAD/T MF0:TEST.BAS/T
/W	Whole Copy or Partial Copy	ED0:=MF0:/W or MF0:=MM0:BASIC.CIL/W
/X	Exit from FUP	/X
/Z	Zero a Directory	ED0:/Z or MF0:/Z or /Z

The FUP program is accessed from the **CONSOLE MONITOR** (CONMON) program, using the following instructions:

When "#" is displayed
1. Type F
2. Press RETURN

In order to leave **FUP** and return to the **CONSOLE MONITOR:**

When * is displayed -
1. Type /X
2. Press RETURN

It is good programming practice to have a second disk and even a third disk with back up copies of your programs. FUP is a powerful tool and its operations are not reversible. If you inadvertently exchange the positions of the source and distination files in a FUP command, you could lose the source file, and in this case, a back up copy prevents a total loss.

John Fluke Mfg. Co., Inc.
P.O. Box 43210, Mountlake Terrace, WA 98043
800-426-0361 (toll free) in most of U.S.A.
206-774-2481 from AK, HI, WA and Canada
206-774-2398 from other countries

Fluke (Holland) B.V.
P.O. Box 5053, 5004 EB, Tilburg, The Netherlands
Tel. (013) 673973, TELEX 52237
Phone or write for the name of your local Fluke representative.

Printed in U.S.A.

B0078B-10U8102/SE EN

FLUKE®

Technical Data

17XXA Highlighted Learning Program B0079

Communication Over The IEEE-488 Bus

IEEE-488

The concept of the **IEEE-488 Bus** allows **MULTIPLE INSTRUMENTS** to be connected to a **COMMON Bus.** Each instrument is assigned a **DEVICE ADDRESS** to allow **SELECTIVE COMMUNICATION** over the bus.

The **PRINT** and **INPUT** statements are the heart of bus communication. They allow **2-WAY COMMUNICATION** to take place between a controller and the various instruments connected to the bus.

A device which is **SENDING DATA** over the bus is called a **TALKER.**

A device which is **RECEIVING DATA** over the bus is called a **LISTENER.**

The general form of the **PRINT** and **INPUT** statements is shown below:

PRINT @ Makes addressed instrument a listener	**2,** Device Address	**"COMMAND STRING"** User assigned Command String enclosed in quotes
INPUT @ Makes addressed instrument a talker	**2,** Device Address	**VARIABLE** User assigned variable for storage of data Example R, AS, T%

An important rule of bus communication is that while there may be **MULTIPLE LISTENERS** on the bus, only **ONE TALKER** may be enabled at any given time.

We will now go through several examples to illustrate the fundamentals of communication over the bus.

In order to do this it will be necessary to use several additional bus commands.

INIT this command initializes the bus to a known state

CLEAR@ the selected device is addressed as a listener and a selective device clear is issued

WAIT Program execution is suspended for the time interval (in milliseconds) specified

Example 1: PROGRAM an 8502A to take a **SINGLE READING** and **DISPLAY** it on the controller.

```
10 INIT PORT 0              initializes bus
20 CLEAR@2                  resets DVM
30 WAIT 5000                waiting to reset
40 PRINT@2, "VR2T0,"        programs DVM for 10V DC range
50 PRINT@2, "?"             triggers reading
60 INPUT@2, R               puts reading from DVM into variable R
70 PRINT R                  displays reading on CRT
```

Example 2: MODIFY the above program to take **10 READINGS** and **DISPLAY** them in **REAL TIME**

add the following program lines to the Example 1 program

45 FOR I% = 1% TO 10%
75 NEXT I%

the complete program is shown below:

```
10 INIT PORT 0
20 CLEAR @ 2
30 WAIT 5000
40 PRINT @ 2, "VR2T0,"
45 FOR I%=1% TO 10%      ┐
50 PRINT @2, "?"         │  FOR NEXT LOOP REPEATS
60 INPUT @ 2, R          │  LINES 50, 60, 70
70 PRINT R               │  10 TIMES
75 NEXT I%               ┘
```

The next example illustrates the important distinction between displaying the readings in **REAL TIME** (Example 2) and displaying them **AFTER** the measurement cycle is complete (Example 3).

EXAMPLE 3: MODIFY the program in Example 2 to **TAKE 1O READINGS** and **STORE** them in **VARIABLE R.** After the **LAST READING** is completed, the program **WAITS 3 SECONDS** and then **DISPLAYS** the **READINGS.**

Add the following program lines to the Example 2 program:

```
5 DIM R (10)
60 INPUT @ 2, R (I%)
70 ! DELETE THIS LINE FROM EXAMPLE 2
80 WAIT 3000
90 FOR I%=1% to 10%
100 PRINT I%; R(I%)
110 NEXT I%
```

The complete program is shown below:

```
5 DIM R(10)                    Dimensions R to hold up to 11 readings
10 INIT PORT 0
20 CLEAR @ 2
30 WAIT 5000
40 PRINT @ 2, "VR2T0"
45 FOR I% = 1% TO 10%          for next loop to take 10 readings
50 PRINT @ 2, "?"
60 INPUT @ 2, R (I%)
75 NEXT I%
80 WAIT 3000                   inserts delay before displaying readings
90 FOR I% = 1% TO 10%
100 PRINT I%; R (I%)           for next loop to display 10 readings
110 NEXT I%
```

Simple as these programs are, the following items were accomplished:

1. Initialize the Bus
2. Clear an instrument
3. Send a command string to an instrument
4. Trigger a reading
5. Store readings in a variable
6. Display readings on the CRT
7. Use a wait statement
8. Dimension a variable
9. Use a for-next loop with an integer variable
10. Use a semicolon to separate printed variables

Note: Additional statements such as RBIN, RBYTE, WBIN, and WBYTE are available to transfer binary data, to speed up data transfer, to overcome data format problems and to control the sequence of data transfers. The software bulletin on the 8500 series digital voltmeter has examples of RBYTE and WBYTE for binary readings and their conversion to floating point format.

FLUKE®

John Fluke Mfg. Co., Inc.
P.O. Box C9090, Everett, WA 98206
800-426-0361 (toll free) in most of U.S.A.
206-356-5400 from AK, HI, WA
206-356-5500 from other countries

Fluke (Holland) B.V.
P.O. Box 5053, 5004 EB, Tilburg, The Netherlands
Tel. (013) 673973, TELEX 52237
Phone or write for the name of your local Fluke representative.

Printed in U.S.A

B0079B-10U8207/SE EN

© 1982, John Fluke Mfg. Co., Inc., all rights reserved

FLUKE®

Technical Data

1720A Highlighted Learning Program B0097

The System Command File

The **System Command File** stores a set of **initialization** instructions that normally would be input from the keyboard after applying power to the controller. It executes the preprogrammed instructions and can call up **BASIC** and run the **main program**. It allows therefore, a set of **start-up** instructions to be executed automatically from the disk.

Use of the **Command File** then, allows the 1720A to be operated completely **independent** of the keyboard.

Some typical **tasks** that would be performed by a **Command File** are shown below.

- CHECK TO SEE WHETHER THE CLOCK HAS BEEN DISTURBED
- SET THE BAUD RATES ON THE RS-232 PORTS
- FORMAT THE ELECTRONIC DISK
- ASSIGN THE SYSTEM DEVICE
- DOWNLOAD FILES FROM THE FLOPPY DISK TO THE ELECTRONIC DISK
- CALL UP BASIC
- EXECUTE IMMEDIATE MODE BASIC COMMANDS
- RUN A BASIC PROGRAM

All modes of the controller are accessible to a **Command File** just as they are from the keyboard.

There are **two ways** to generate a **System Command File:** It can be entered directly from the **keyboard** while in **FUP**, or a **BASIC program** can be used to read **data statements** into the file. Using the **BASIC program** allows the command file to be easily modified using the **BASIC editor.** For simple command files the **FUP** method is quicker. However once a command line is entered it cannot be corrected unless the whole command file is rewritten.

Generating A System Command File Using FUP:

1. From **CONMON** enter **"F"** to select the **File Utility Program.**
2. Enter **MFO:COMND.SYS=KBO:** This will assign the file **"COMMND.SYS"** on the floppy disk **(MFO:)** as the destination for source information to come from the keyboard.
3. Enter each **command** to be **stored** on a **separate line.** Use the **delete** key to correct for errors before terminating the line with a (CR).
4. Terminate the file with a **CTRL/2** (end of file).

AN EXAMPLE IS SHOWN BELOW:

Console Monitor Version x.y 8:52 19-JUN-80

```
* F (CR)
```

FUP 8:52 19-JUN-80
File Utility Program Version x.y

```
* MFO:COMMND.SYS=KBO:
T          (Calls TIME)
F          (Calls FUP)
EDO:/F     (Formats E-Disk)
Y          (YES)
/X         (Exit from FUP)
B          (Calls BASIC)
RUN"METER.BAS"
  Z        (End of File)
```

Generating a System Command File Using BASIC

This program creates two files on the floppy disk: a Command File under the name **COMMD. SYS,** and a copy of this program under the name **COMMD.BAS.** The DATA statements (except line 10) contain the actual command file. Use as many DATA statements as needed. Each **data** statement must contain **one** legitimate keyboard command. All modes of the 1720A are accessible to a Command File just as they are from the keyboard.

1. From **CONMON**, type **B** to select the BASIC Interpreter.
2. Type **EDIT** to select the **edit mode** of BASIC.
3. Now enter the following program using the **editing capabilities** of BASIC. Use as many **data statements** as needed, with one quoted command line in each statement.

```
1000    ON ERROR GOTO 10060
1010    DATA "  (COMAND     "
1020    DATA "  Inputs      "
1030    DATA "  For         "
1040    DATA "  CONMON      "
1050    DATA "  SET         "
1060    DATA "  FUP,        "
1070    DATA "  BASIC       "
1080    DATA "  ETC.)       "
.
.
(More lines of DATA, as needed)
.
.
10000   DATA "END"
10010   OPEN "MFO:COMMND.SYS" AS NEW FILE 1
10020   READ A$\ IF A$="END" GOTO 10040
10030   PRINT #1,A$\ GOTO 10020
10040   CLOSE 1
10050   KILL "MFO:COMMND.BAS"\SAVE "MFO:COMMND"\GOTO 10070
10060   IF ERL=10050 AND ERR=305 THEN SAVE "MFO:COMMND" ELSE OFF ERROR
10070   END
```

Hints On Using Command Files

* Several **Command Files** can be kept on a disk and one made **active** by **assigning** it to **COMMND.SYS.** Assume we have the following command files stored on disk.

COMMND.ONE
COMMND.TWO
COMMND.TMP

We can make **COMMND.ONE** the **Active Command File** by using **FUP** as shown below.

EXAMPLE:

***MFO:COMMND.SYS=MFO:COMMND.ONE (CR)**

CAUTION: *This will delete whatever was previously stored as "COMMND.SYS."*

This creates a file **COMMND.SYS** on the disk using the contents of **COMMND.ONE.** Note that **COMMND.ONE** is still retained on the disk for backup purposes.

Similarly, to **deactivate** the Command File, just delete **COMMND.SYS** as shown below.

EXAMPLE:

***MFO:COMMND.SYS/D (CR)**

A useful application of this would be where **COMMND.SYS** on the disk is used to **download** the contents of the **Disk** to **E-Disk, assign** the E-Disk as the **system device, assign** a file from the disk (**COMMND.EDO:**) as the Commnd File for E-Disk, and then **run a user program.** The two files as they would appear on the disk are shown below.

```
*COMMND.SYS (CR)
T
F
EDO:/F
Y
EDO:=MFO:/W
EDO:/A
COMMND.SYS=COMMND.EDO:/R
/X
B
RUN "USER.PRG"
```

1980, John Fluke Mfg. Co., Inc., all rights reserved

```
*COMMND.EDO: (CR)
T
B
RUN "USER.PRG"
*DISPLAYING THE CONTENTS OF THE
COMMAND FILE WHILE IT IS ACTIVE
```

Often it is useful to **display** the **instructions** contained in the **Command File** while it is active. This can be easily accomplished by adding an instruction to the Command File telling it to **list COMMND.SYS** while in **FUP.** This is illustrated in the sample Command File shown below.

EXAMPLE:

```
*COMMND.SYS (CR)
T
F
COMMND.SYS (Displays the File)
EDO:/F
Y
B
RUN"TEST 1.BAS"
*DISPLAYING MESSAGES WHILE THE
COMMAND FILE IS ACTIVE
```

If instead of displaying the contents of **COMMND.SYS** as was done above, we can display the contents of a different file. This allows us to **display a message** by putting it in the file. We will call the file **DSPLY.MSG.** It can be **generated** from **FUP** just as Command Files can.

EXAMPLE:

```
*DSPLY.MSG=KBO:
  (CR)
  (CR)
          PLEASE STANDBY!!!!
    DOWNLOADING FLOPPY TO E-DISK
  (CR)
  Z
```

It is possible to make use of the **attributes** capability of the 1720A when generating the **Display File** but that procedure will be covered in a later lesson.

To use the **Display File** merely place the instruction to display it in the command file as shown below.

EXAMPLE:

```
*COMMND.SYS (CR)
T
F
DSPLY.MSG (Displays Message)
EDO:/F
Y
EDO:=MFO:/W
EDO:/A
/X
B
RUN"USER.PRG"
```

John Fluke Mfg. Co., Inc.
P.O. Box C9090, Everett, WA 98206
800-426-0361 (toll free) in most of U.S.A.
206-356-5400 from AK, HI, WA
206-356-5500 from other countries

Fluke (Holland) B.V.
P.O. Box 5053, 5004 EB, Tilburg, The Netherlands
Tel. (013) 673973, TELEX 52237
Phone or write for the name of your local Fluke representative.

©1982, John Fluke Mfg. Co., Inc., all rights reserved

Printed in U.S.A.

B0097A-10U8009/SE EN

FLUKE®

Technical Data

17XXA Highlighted Learning Program B0098

Disk Initialization

FUP, the File Utility Program, is a file transfer and management program giving you flexible control over the files on floppy disks and on the optional Electronic Disk. FUP sets up a communications channel between devices and files recognized by the **Floppy Disk Operating System (FDOS).**

/F - FORMAT A DISK OR ELECTRONIC DISK

Before a floppy disk is used for the first time, its magnetic surface is completely blank. In this form it is not possible for the 1720A to write on the floppy disk until it has been **FORMATTED.** The formatting procedure writes addresses in the form of tracks and sectors onto the floppy disk so that specific locations may be referenced by the 1720A. FUP is used to format either the floppy disk or Electronic Disk.

SYNTAX: **[<DEVICE>], <DEVICE>]]/F<CR>**
DEFAULT: **<DEVICE>=SYO**

While multiple device formatting is possible we will deal with formatting one device.

This reduces the command line to:

[<DEVICE>] / F <CR>

If we wish to format a floppy disk we would insert a blank disk in the drive and use the following command line:

***MFO: / F <CR>**

But if MFO: is the default device (MFO: is SYO:) the command line would be:

Example:

```
*/F <CR>
Really zero SYO:? YES <CR>
```

Notice that the 1720A asks for CONFIRMATION OF THE FORMATTING COMMAND. After **FORMATTING,** the **DEVICE DIRECTORY** will be **ZEROED.**

NOTE:** If a device containing **FILES** is **FORMATTED,** all **FILES** on the device are **ELIMINATED (erased).

Similarly, if MFO: is SYO: and we want to **FORMAT** the **ELECTRONIC DISK** we would use the following command line.

Example:

```
*EDO: F <CR>
Really zero EDO:? YES <CR>
```

/S - BAD BLOCK SEARCH

This command scans devices for **BAD BLOCKS.** It will print to the **OUTPUT FILE** the **NUMBER** of the **BAD BLOCK** and the **TOTAL NUMBER OF BAD BLOCKS** found. In addition, if any of the blocks have **SOFT ERRORS** (requiring more than one try to write or read valid information onto the block) it will indicate the **NUMBER** of **RETRIES** that were necessary.

SYNTAX: **[<OUTPUTFILE> =][<DEVICE>[,**
<DEVICE>]] / S <CR>

DEFAULT: <DEVICE> = SYO <OUTPUT FILE>
= KB0:

While it is possible to scan multiple devices for bad blocks we will deal with one device. This reduces the command line to:

[<OUTPUTFILE> =][<DEVICE>] / S <CR>

If we wish to scan a disk the command line would be:

***KBO: = MFO: / S<CR>**

But since **KBO:** is a default condition and if **MFO:** is a default condition the command line reduces to **/ S <CR>**

Example:

```
*/ S <CR>

Block 273 has 1 entry
Block 275 has 1 retry
Block 326 has 3 retries
Block 351 is bad
Block 358 is bad

Total of 2 bad blocks found
```

Similarly if **EDO: is SYO:** we could scan a disk **(MFO:)** and print the diagnostics to a **FILE** on **EDO** called **SCAN.BLK.** The command line would be:

***EDO:SCAN.BLK=MFO: / S<CR>**

But since EDO: is the default device the command line would reduce to:

***SCAN.BLK=MFO: / S<CR>**

/L - List the DIRECTORY

This command prints the directory of the specified **DEVICE** to the specified **OUTPUTFILE.**

Syntax: **[<OUTPUTFILE>]=[<DEVICE>]/L<CR>**

Default: **<OUTPUTFILE>** = **KBO** (console device)
<DEVICE> = **SYO** (system device)

If we wish to list the directory of a floppy disk on the display **(KBO:),** the command would be:

***KBO: = MFO: /L <CR>**

but since **KBO:** is the default condition and if **MFO:** is the default condition, the command simplifies to **/L <CR>**

Example:

***/L <CR>**

Directory of SYO: on 3-Oct-79 at 16:34

Name.Ext	Size	Date
FDOS.SYS	13	25-Sep-79
CONMON.SYS	2	20-Sep-79
TIME.CIL	2	20-Sep-79
SET.CIL	1	20-Sep-79
FUP.CIL	9	20-Sep-79
BASIC.CIL	50	20-Sep-79
EDIT.CIL	9	20-Sep-79
SELECT.BAS	12	20-Sep-79
DEMO.BAS	46	21-Sep-79
LIST.13E	5	25-Sep-79
IEEE.BAS	1	2-Oct-79
ENTER.BAS	4	3-Oct-79
8520.1	29	3-Oct-79
8520.BAS	29	3-Oct-79

Total of 212 blocks in 14 files, 186 free blocks

Similarly, if we want to list the above directory out to a printer connected to the RS-232 port **KB1:,** the following command would be used:

***KB1: =MFO:/L <CR>**

And due to the default conditions it would simplify to:

***KB1: =/L <CR>**

Finally, if **SYO:** is **MFO:** and we wish to list on the display the directory contained on electronic disk, the command would be:

***KBO:=EDO:/L <CR>**

And due to the default conditions we would use:

***EDO:/L </CR>**

/E - EXTENDED DIRECTORY listing

Since the 1720A employes a **CONTIGUOUS file** structure, there can be blocks of **UNUSED ENTRIES.** These won't be displayed with the **/L** command. If a listing of these empty entries is desired, however, the **/E** command can be used instead of the **/L** command.

Example:

***/E <CR>**

Directory of SYO: or 3-Oct-79 at 16:34

NAME.EXT	SIZE	DATE
FDOS.SYS	13	25-Sep-79
CONMON.SYS	2	20-Sep-79
TIME.CIL	2	20-Sep-79
SET.CIL	1	20-Sep-79
FUP.CIL	9	20-Sep-79
BASIC.CIL	50	20-Sep-79
<NOT USED>	30	
DEMO.BAS	46	21-Sep-79
<NOT USED>	5	
ENTER.BAS	4	3-Oct-79
<NOT USED>	236	

Total of 127 blocks in 8 files, 271 free blocks

1982, John Fluke Mfg. Co., Inc., all rights reserved

The angle brackets indicate blocks of **UNUSED ENTRIES** located between files. The **/E** command then indicates not only the amount of unused blocks but also where they are distributed on the disk.

/P - PACK A DEVICE

In order to best utilize the unused blocks on a disk it is often desirous to have all the free space on a disk in one **Contiguous Section** located after the existing files. This is accomplished with the **/P** command.

SYNTAX: [<**DEVICE**> [, <DEVICE>]] **/P** <**CR**>

DEFAULT: <**DEVICE**> = **SYO**

While multiple device packing is possible we will deal with packing one device. This reduces the command line to:

[<DEVICE>] /P <CR>

Here's how we take the disk contents from the previous **/E DIRECTORY LISTING** and pack them. The command line would be:

***MFO:/P <CR>**

But if **MFO:** is the default device the command line reduces to **/P <CR>.**

Example:

***/P <CR>**

***/E <CR>**

Directory of SYO: on 3-Oct-79 at 16:40

NAME.EXT	SIZE	DATE
FDOS.SYS	13	25-Sep-79
CONMON.SYS	2	20-Sep-79
TIME.CIL	2	20-Sep-79
SET.CIL	1	20-Sep-79
FUP.CIL	9	20-Sep-79
BASIC.CIL	56	20-Sep-79
DEMO.BAS	46	21-Sep-79
ENTER.BAS	4	3-Oct-79
<NOT USED>	271	3-Oct-79

Total of 127 blocks in 8 files, 271 free blocks.

Notice that now the all free space on the disk is located in one section and there are no unused entries between files.

If we want to pack the **ELECTRONIC DISK** when **SYO:** is the floppy disk **(MFO:)** we would use the following command line:

***EDO:/P <CR>**

John Fluke Mfg. Co., Inc.
P.O. Box C9090, Everett, WA 98206
800-426-0361 (toll free) in most of U.S.A.
206-356-5400 from AK, HI, WA
206-356-5500 from other countries

Fluke (Holland) B.V.
P.O. Box 5053, 5004 EB, Tilburg, The Netherlands
Tel. (013) 673973, TELEX 52237
Phone or write for the name of your local Fluke representative.

Printed in U.S.A.

B0098B-10U8208/SE EN

FLUKE®

Technical Data

17XX Highlighted Learning Program B0099

Modification of Files

FUP, the File Utility Program, is a file transfer and management program giving you flexible control over the files on floppy disks and on the optional Electronic Disk. FUP sets up a communications channel between devices and files recognized by the **Fluke Disk Operating System (FDOS).**

FUP can be used to easily accomplish the following file modifications:

/R RENAME A FILE

SYNTAX: <DESTINATION FILE> = <SOURCE FILE>
/R <CR>

Default: **None**

The **INPUT FILE** is **RENAMED** according to the name as specified in the **OUTPUT FILE** list.

Example:

To take the file **193A.BAS** and **RENAME** it to **1953A.BAS** we would use the following command line:

***1953A = 193A/R <CR>**

NOTE: *If the devices don't correspond, an error message is given. If the output file already exists, an error message is given.*

/D DELETE A FILE

This command **DELETES** the specified file or files.

SYNTAX: **<FILE NAME> [, <FILE NAME>]**
/D <CR>

DEFAULT: None

Example:

If it is desired to **DELETE** the files **METER.BAS** and **TEST.488** the following command line would be used:

***METER, TEST.488/D <CR>**

NOTE: *Limited to eight files at a time.*

/M MERGING ASCII FILES

SYNTAX: [<DESTINATION FILE> =]<SOURCE FILE>[,
<SOURCE FILE>]/M <CR>

DEFAULT: **<DESTINATION FILE> = KBO:**
(CONSOLE DEVICE)

CAUTION: *Line numbers of the files to be merged should not overlap.*

Example:

To MERGE the content of **OLD.BAS** and **NEW.BAS** and create a new file **NEW.BAS** all on the system device we would use the following command line:

***NEW = OLD, NEW/M <CR>**

NOTE: *There can be a maximum of* ***8 FILES*** *specified in the* ***SOURCE*** *(or right side of the command).*

ERROR FREE FUP

FUP is a powerful tool. Good programming practices should be employed to avoid pitfalls when using it.

1. Make a back up copy of your disk before modifying or packing any files. If you do make an error afterwards, it won't be catastrophic.
2. Verify the results of each FUP operation you do. View the directory or files as necessary to accomplish this. It is better to discover if you have inadvertently erased a file or disk at the finish of each operation than when the program can't be recovered.
3. Verify your FUP commands as typed before you execute them. Imagine what would happen if you typed MFO:/Z when you really meant to type MMO:/Z.
4. Adequately identify your Floppy disks by writing something meaningful on the label with a felt tip pen. Formatting an already recorded disk which has a blank label can ruin your day. Better yet, do a /L command on a "blank" disk before you format it. If you get a directory listing, you have saved a disk.
5. Routinely Pack and Search your disks for bad blocks. Disks which have bad blocks should have their files copied onto another disk. If re-formatting the questionable disk doesn't get rid of the bad blocks, the disk should be discarded.
6. Be sure to use /B or /W when transferring binary files. /B will handle any ASCII files as well.
7. Be sure to store your system command file under a temporary name before creating a new command file.

ERROR MESSAGES

Listed below are the error messages and their explanations which can be encountered in FUP.

DEVICE ERROR
This error indicates a fatal error on a I/O device. Examples are disk CRC errors, magnetic tape parity errors.

?DEVICE NOT READY
This means that the accessed device is not ready. Remedies are loading a diskette or closing the disk door.

?WRITE PROTECTED
The diskette has a write protect tab. This can be fixed by either removing the tab or using a non protected diskette.

?NOT A VALID DEVICE
This means that a device has been specified for which there is no support in FDOS. Misspelling is often the cause.

?FILE NOT FOUND
This indicates that the required file could not be found on the specified device.

?NO ROOM FOR USER ON DEVICE
This indicates that the created file caused by either merging or copying exceeds the amount of available space.

?NO END-OF-FILE
An ASCII file had been used as an input file, while not containing an End-Of-File mark.

?DEVICES DO NOT MATCH
A rename option with different devices has been given.

?NOT A FILE STRUCTURED DEVICE
Certain operations like packing or listing a directory are legal only on file structured devices.

?SYNTAX ERROR
A command line as input by the user did not meet the syntax specification as required for that particular command.

?TOO MANY FILES
More than eight files had been specified either in the input or output file specification.

?ILLEGAL OPTION
An unrecognized option was selected. Probably a typing error.

?NOT A VALID FILE NAME
A filename with illegal or too many characters had been entered.

USING FUP TO DE-BUG BASIC PROGRAMS

Files which have been OPENED for the storage of ASCII data can be viewed via FUP by typing the file name followed by the carriage return. If it is a long file the paging keys should be used. Data is stored on the disk by the PRINT command in identically the same format that is used to PRINT via a mechanical printer.

The data being stored on the disk can be considered as a printed page. In this context, errors such as "input line too long", "too much data typed" and "not enough data typed" become meaningful.

If your basic program is having problems outputing data to the disk or inputing data from the disk, viewing the data file on the CRT via FUP can provide significant insight.

John Fluke Mfg. Co., Inc.
P.O. Box C9090, Everett, WA 98206
800-426-0361 (toll free) in most of U.S.A.
206-356-5400 from AK, HI, WA
206-356-5500 from other countries

Fluke (Holland) B.V.
P.O. Box 5053, 5004 EB, Tilburg, The Netherlands
Tel. (013) 673973, TELEX 52237
Phone or write for the name of your local Fluke representative.

Printed in U.S.A.

B0099B-10U8208/SE EN

© 1982, John Fluke Mfg. Co., Inc., all rights reserved

FLUKE®

Technical Data

17XXA Highlighted Learning Program B0100

Copying (Transferring) of Files

The File Utility Program is the ideal program to use for duplication of files and disks. FUP inputs information directly from the original disk to the output device **For Temporary Storage** for writing on the disk you are duplicating. More than one file can be duplicated at a time; in fact, it is possible to duplicate an entire disk with one command, providing the original does not exceed 89 blocks when MMO: is the destination or 254 blocks when EDO: is the destination 508 blocks if 2 E-Disks are used).

ZERO THE DIRECTORY

There are two reasons for zeroing a directory. In the first case, you have a floppy disk or E-Disk you wish to erase. In the second case, you are using **EDO:** or **MMO:** as **TEMPORARY STORAGE** for duplicating floppy disks (floppy disk #1 to temporary storage; temporary storage to floppy disk #2) and the original floppy disk contains more blocks than the temporary storage. Consequently, the temporary storage must be copied to more than once and its directory must be zeroed before it can be copied to again.

When the **/Z** command is used, the **DIRECTORY** of the specified **DEVICE** is **ZEROED OUT,** that is, **ALL FILE HEADERS WILL BE WIPED OUT.** (Note that the file itself still exists, just the pointer, or file header is erased.)

SYNTAX: = **[<DEVICE>** [, <DEVICE>]]**/Z <CR>**

DEFAULT: <**DEVICE**> = **SYO**

While multiple device zeroing is possible we will deal with zeroing one device. This reduces the command line to:

```
= [<DEVICE> ]/Z <CR>
```

If we wish to zero the directory on a disk we would use the following command line:

```
*MFO:/Z <CR>
```

But if MFO: is the default device the command line would be:

```
*/Z <CR>
Really zero SYO.? YES <CR>
```

Notice that the 1720A asks for **CONFIRMATION** of the **ZEROING COMMAND.**

Similarly if **MFO:** is **SYO:** and we want to **ZERO** the **DIRECTORY** on **ELECTRONIC DISK** we would use the following command line:

```
*EDO:/Z <CR>
Really zero EDO: ? YES<CR>
*
```

If a device contains a **DIRECTORY** it had to have been previously **FORMATTED.** Therefore after **ZEROING** an **EXISTING DIRECTORY,** the device **DOES NOT** require **FORMATTING** in order to use it again.

/W - WHOLE COPY A DEVICE

The **/W** command allows **FAST DUPLICATION OF DISKS.**

All files from the **SOURCE DEVICE,** starting at the specified **START FILE,** are **COPIED** to the **DESTINATION DEVICE** until no more exist or the destination device is full.

SYNTAX: **[<DESTINATION DEVICE>] = [SOURCE DEVICE>** [<START FILE>]]**/W <CR>**

DEFAULT: **<DESTINATION DEVICE> : <SOURCE DEVICE> = SYO**

If **NO START FILE** is specified, the **WHOLE DEVICE** is copied. The copy process continues until there is **NO ROOM** on the **OUTPUT DEVICE** for another file or until **ALL FILES** have been **COPIED.** If a **CTRLC** is given during a file transfer, the copy process is **ABORTED** when copying of the **CURRENT FILE** has been completed.

When a **CRTL P** is given, the copy process is **ABORTED** and the **CURRENT FILE** will be **CLOSED** even though the **WHOLE FILE MIGHT NOT** have been **TRANSFERRED.** The names of all the copied files are displayed on the console terminal.

If we wanted to **COPY** the contents of a **DISK** to **ELECTRONIC DISK** we would use the following command line:

```
*EDO: = MFO:/W <CR>
```

But if **MFO:** is the default device the command line would be:

```
*EDO: = /W <CR>

Copying   FDOS.SYS
Copying   CONMON.SYS
Copying   TIME.CIL
Copying   SET.CIL
Copying   FUP.CIL
Copying   BASIC.CIL

*
```

When the * is displayed, it indicates the transfer is completed and the contents of **MFO:** are now in **EDO:**

If we wished to make a **DUPLICATE COPY** of the disk we have just transferred, we merely have to insert another **FORMATTED** disk in the drive and transfer the **FILES** back from **EDO:**. The command line would be:

MFO: = EDO:/W <CR>

But since **MFO:** is the default device the command line would be:

```
*= EDO:/W <CR>

Copying   FDOS.SYS
Copying   CONMON.SYS
Copying   TIME.CIL
Copying   SET.CIL
Copying   FUP.CIL
Copying   BASIC.CIL

*
```

When the * is displayed it indicates the transfer is completed and the contents of **EDO:** are now on the **NEW DISK** which had been inserted in drive (**MFO:**).

Since a disk can store **200K BYTES** of information and one **ELECTRONIC DISK** can store **128K BYTES** it would require **2 PASSES** to duplicate a disk with more than **128K BYTES (DUAL ELECTRONIC DISKS would eliminate this problem as there would be 256K BYTES** available).

We will assume that the disk files start with **FDOS.SYS,** and the **SECOND PASS** will copy the remaining files starting with the one after **8520A.BAS** which we will assume to be **NEW.BAS** and end with the last file **TEST.BAS.**

The procedure is as follows: (Assume **MFO: = SYO:**)

FIRST PASS

Insert **DISK #1** (the disk to be copied) in the drive.

```
* EDO: = /W <CR>
Copying FDOS.SYS
    ↓       ↓
Copying 8520A.BAS
No room for user on device
*
```

Notice that the 1720A indicates when there is no more room on the device to copy any additional files. Remove **DISK #1** and insert **DISK #2** (the duplicate disk) in the drive.

```
* = EDO:/W <CR>
Copying FDOS.SYS
    ↓       ↓
Copying 8520A.BAS
*
```

SECOND PASS

Since there is no more room on **EDO:** it is necessary to **ZERO** its **DIRECTORY** before transferring the remaining files to it.

```
* EDO:/Z <CR>
Really Zero: ? YES<CR>
*
```

Now remove **DISK #2** and insert **DISK #1** in the drive.

Since we have already partially copied the files on **DISK #1** it is now necessary to **SPECIFY** a **START FILE (NEW.BAS)** in order to transfer the remaining files to **EDO:**.

```
* EDO: = NEW.BAS/W <CR>
Copying NEW.BAS
    ↓       ↓
Copying TEST.BAS
*
```

©1982, John Fluke Mfg. Co., Inc., all rights reserved

At this point all the remaining files have been transferred to **EDO:.**

Remove **DISK #1** and insert **DISK #2** in the drive.

```
* = EDO:/W <CR>
Copying NEW.BAS
Copying TEST.BAS
*
```

At this point, all the files on **DISK #1** have been copied onto **DISK #2.**

COPY OF FILES USING MAIN MEMORY (MMO:)

If a 1720A is not equipped with **ELECTRONIC DISK** it is still possible to copy files as was previously done with **EDO:** by using the **MAIN MEMORY.**

FUP can assign all the available **MAIN MEMORY** to act as a **MASS STORAGE DEVICE.** A special device will be created by **FUP** to accommodate this function. This device (**MMO:**) is only valid in **FUP** and is main memory. Its size depends on the amount of available main memory. Using the **TRANSFER** or **WHOLE COPY** commands, the user can fill this device with files from the disk, change the diskette and copy form **MMO:** to the disk. (See **TRANSFERRING BINARY FILES** and **TRANSFERRING ASCII FILES** on the following pages.) If **MMO:** can not hold all the files, the user must **PERFORM ADDITIONAL PASSES** in order to copy the remainder of the files. Note that **MMO:** will be automatically zeroed when **FUP** is started.

CAUTION: Certain operations like formatting a floppy disk will cause MMO: to be zeroed.

Example:

```
* MMO: = /W
Copying FDOS.SYS
Copying CONMON.SYS
Copying BASIC.CIL
*
```

NOTE: MMO: does **not** require formatting.

At this point all files have been transferred and the user must now swap disks.

```
* = MMO:/W
Copying FDOS.SYS
Copying CONMON.SYS
Copying BASIC.CIL
*
```

REMEMBER, as when using **EDO:** and making **MULTIPLE PASSES** it is necessary to **ZERO THE DIRECTORY** on **MMO:** and **SPECIFY A START FILE** after the **FIRST PASS.**

LIST ASCII FILES

SYNTAX: **[<DESTINATION FILE> = <SOURCE FILE> <CR>**

DEFAULT: **<DESTINATION DEVICE>=SYO:** if a **<FILENAME>** is specified

<DESTINATION FILE> = KBO: if **<DESTINATION FILE>** is omitted.

If **NO DESTINATION FILE** is **SPECIFIED** the **SOURCE FILE(S)** will be printed on the Console Terminal. This gives us a convenient way to list **ASCII** files on the Console Terminal.

Example:

```
*METER.488 <CR>
```

will cause the file to be listed on the Console Terminal.

Example:

```
* KB1: = METER.488 <CR>
```

will cause the file to list out to a printer connected to RS-232 port KB1:

TRANSFERRING BINARY FILES

In order to transfer a BINARY FILE it is necessary to use the **/B** command.

For instance if **MFO:** = **SYO:** and we wish to transfer the file **BASIC.CIL** over to **ELECTRONIC DISK** we would use the following command line:

```
* EDO:= BASIC.CIL/B <CR>
```

NOTE: *Binary transfers can be made for all types of files, (ASCII, etc.).*

TRANSFERRING ASCII FILES

If **MFO:** = **SYO:** and we wish to transfer the files **METER.BAS** and **NEW.TST** from the disk over to **ELECTRONIC DISK** we would use the following line:

```
* EDO: = METER, NEW.TST <CR>
```

/T TURN OFF ERROR CHECKING AND TRANSFER/COPY AN ASCII FILE

SYNTAX: Identical to copy command

On those rare occasions when one or more blocks in a file have gone bad, the user may want to recover as much of the file as possible. They can do so by defeating some error checks. (Device error and No End-of-file) If a device error occurs the block will be transferred as is, that is, it may contain garbage. If an end-of-file mark has disappeared a new one will be automatically appended.

Example:

```
* EDO: = METER.488/T <CR>
```

John Fluke Mfg. Co., Inc.
P.O. Box C9090, Everett, WA 98206
800-426-0361 (toll free) in most of U.S.A.
206-356-5400 from AK, HI, WA
206-356-5500 from other countries

Fluke (Holland) B.V.
P.O. Box 5053, 5004 EB, Tilburg, The Netherlands
Tel. (013) 673973, TELEX 52237
Phone or write for the name of your local Fluke representative.

Printed in U.S.A.

B0100B-10U8208/SE EN

FLUKE®

Technical Data

1720A Highlighted Learning Program B0123
E-DISK ™

The optional electronic disk is a high-speed, file-structured mass storage system. It works just like floppy disk, differing only in capacity and speed. E-Disk files can easily be transferred to a floppy disk for permanent storage. When installed, the E-Disk occupies one or both of the circuit module slots marked "SPARE".

FEATURES

- High speed
- Large capacity
- Works just like the floppy disk
- Battery backup available for power interruptions

HIGH SPEED

The speed of a file storage system is normally described in two ways:

1. Access time: The time lag between identifying a file request and the beginning of file transfer.
2. Transfer rate: The rate that data is transferred after it has been accessed.

Table 1 compares the hardware speed performance of the E-Disk with that of the floppy disk. Note that software processing overhead is in addition to these times.

	E-DISK	FLOPPY DISK	E-DISK ADVANTAGE
ACCESS TIME (milliseconds)	.018	550 (average)	30,600:1
TRANSFER RATE (bytes/second)	133,120	15,625	8.5:1
TIME TO FILL MAIN MEMORY (seconds)	0.46	3.9	8.5:1

Among the benefits of this high speed are:

* Most lexically-saved forms of BASIC programs load so quickly that the short delay time for loading is not apparent to an operator.
* Sequential files and random access virtual arrays can be built up and accessed within a running program without significantly slowing it down.

LARGE CAPACITY

Programs and data files on the E-Disk are organized as a sequence of 256 or 512 data blocks. Each data block contains 512 bytes of information. The first two blocks (1K bytes) are reserved for a file directory.

A second E-Disk module doubles storage capacity without the need to address each module separately. Because of this, a single large file can occupy two modules.

The largest file that a single module E-Disk can store is 254 blocks (127K bytes). A dual module system can store 510 blocks (255K bytes). The maximum number of files is 72.

Some of the benefits of the large capacity of an E-Disk are:

* Sequential files and random access virtual arrays can be much larger than would be possible in main memory without significant sacrifice in processing speed.
* Large programs that will not fit into main memory can be structured into modules, and chained together. Although this can be done with the floppy disk, use of the E-Disk nearly eliminates the delays of loading each program module into memory, and frees the floppy disk for collection of processed data.
* Floppy disk files that are too large to fit in main memory can be duplicated. (A dual-module system is required for files larges than 127K bytes.)

WORKS JUST LIKE THE FLOPPY DISK

Techniques for using the E-Disk are identical to the floppy disk. Like any disk, the E-Disk must be formatted before use, and contains a file directory.

The E-Disk is referred to as EDØ: in program statements and utility commands.

When installed, the E-Disk can easily be set up to be the default System Device for files not given a named location.

Some of the benefits of working like the floppy disk are:

* Programming is simplified. Taking advantage of the E-Disk requires no new techniques. The only difference is referring to EDØ: (the E-Disk) in place of MFØ: (the floppy disk).
* By making use of the System Device concept, programs can be structured to run on any 1720A system, taking advantage of the E-Disk whenever it is installed. In such a case, an E-Disk will increase the speed of processing sequential and virtual array files, and eliminate some requirements for an operator to exchange disks.

E-Disk ™ is a trademark of the John Fluke Mfg. Co., Inc.

BATTERY BACKUP

The E-Disk will hold its contents intact when power is removed, provided the E-DISK BATTERY switch on the rear power supply panel is set to ENABLE, and there is an internal or external source of battery power. A fully-charged internal battery will support a single-module E-Disk for about one hour, or a dual-module E-Disk for about 30 minutes. An external battery can easily be connected through the rear REMOTE INTERFACE connector to extend this time indefinitely.

If you do not plan to make use of the battery backup feature for the E-Disk, battery life can be extended by leaving the feature disabled. This will prevent the battery from having to discharge and recharge regularly.

SOME EXAMPLES

1. This Immediate Mode BASIC command will store the lexical form of the program currently in main memory on the E-Disk, in a file named "DEMO.BAL":

   ```
   SAVEL "ED0: DEMO"
   ```

2. This FUP command copies a file from the floppy disk to the E-Disk:

   ```
   ED0:DATA.T9=MF0:DATA.T9
   ```

3. The following FUP command is identical to the previous one, if the E-Disk is the default System Device. The file name is carried over unchanged when it is not specified:

   ```
   =MF0:DATA.T9
   ```

4. If a different floppy disk is then inserted, the following FUP command writes a duplicate copy of the file onto the second disk from the copy on the E-Disk. This also assumes the E-Disk is the System Device:

   ```
   MF0:=DATA.T9
   ```

5. When a program uses a data file on the E-Disk, a channel must first be opened. Input and output statements then refer to the channel number. The following Fluke BASIC sequence opens an existing file on the E-Disk for sequential input, and then reads its contents into the string M3$:

   ```
   2400 OPEN "ED0:DATA.T8" AS FILE 4
   2410 INPUT #4, M3$
   ```

6. The following Fluke BASIC sequence opens a new file on the default System Device for sequential output, and then stores a message in it. When the E-Disk is installed, this file will normally go onto it unless the System Device was reassigned:

   ```
   3920 OPEN "MSG4" AS NEW FILE 2 SIZE 1
   3930 PRINT #2, "End of Frequency Test"
   ```

7. A random access virtual array requires a dimension statement referencing a previously opened channel. The following sequence opens a random access (DIM) file on the E-Disk, dimensions an integer array into it, and then places a value stored in integer variable N% into one of the array elements. Note that virtual array channels are bidirectional. The NEW specification causes the array elements to be initially set to zero, replacing any existing file found with the same name. Virtual arrays are discussed in a separate HLP bulletin.

   ```
   1020 OPEN "ED0: DATA.43" AS NEW DIM FILE
        3 SIZE 1
   1030 DIM #3, D% (15,31)              !512 elements
   .
   .
   .
   4960 D% (4,6) = N%                   !N% Previously defined
   ```

E-DISK TERMINOLOGY

Access Time
The time lag between identifying a file request and the beginning of file transfer.

ASCII
The American Standard Code for Information Interchange is a set of defined 7-bit binary code patterns representing the full alphabet, numbers, and many useful symbols and control characters.

Bit
A single unit of program or data information, set to either one or zero. Bit is a contraction of binary digit.

Block
A fixed size of data selected to be convenient for storage or transfer operations. In the 1720A Instrument Controller, a block is 512 bytes.

Block-Structured
A block-structured device transfers and stores data one block at a time, accumulating smaller amounts in a temporary buffer until a block is available. E-Disk is a block-structured device.

Buffer
A temporary memory storage area for accumulating information until enough is available for the next operation. E-Disk transfers use a 1-block (512-byte) buffer.

©1980, John Fluke Mfg. Co., Inc., all rights reserved

Byte
Eight data bits set to any pattern of ones and zeros with defined meaning, such as binary numbers or the ASCII code.

Chaining
The technique of separating a program into task-oriented modules. Each module contains a RUN "next module" statement linking it to the next task. Variables can be passed to the next program module through a common storage area, and data in virtual arrays can be left open for subsequent use. Only the module currently in use occupies space in main memory.

E-Disk ™
The optional electronic disk is a mass file-storage system constructed of high-speed, solid-state memory and designed to function as a serial transfer block-structured device. With appropriate operating system software, it appears to the system as a file-structured device.

File
A collection of information designated by name as a unit.

File-Structured
A file-structured device transfers and stores information by file units.

Formatting
Formatting is a process of preparing a file-storage device to accept files. After verifying the integrity of each block, an identifying and timing pattern is recorded throughout. The 1720A Instrument Controller accomplishes this with the /F command option of the FUP Utility Program.

K Byte
1024 bytes.

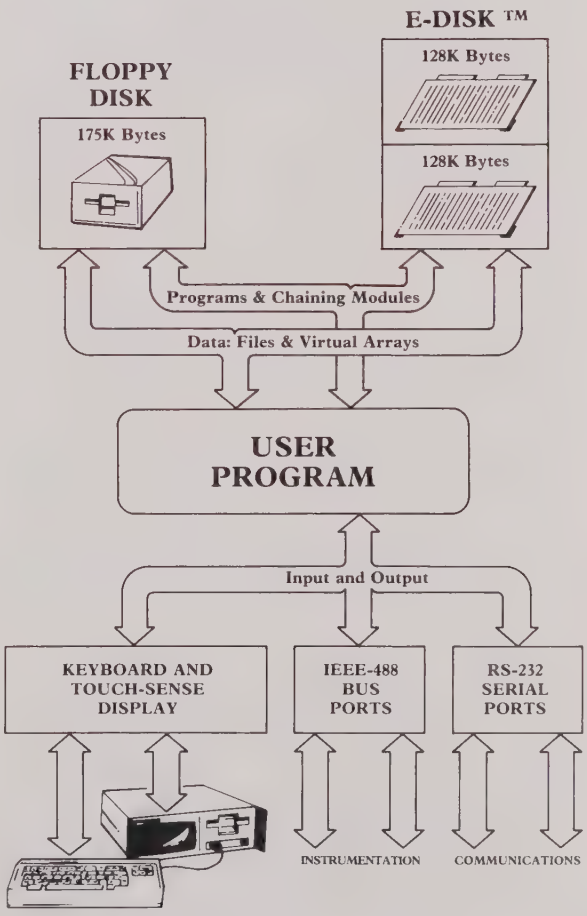
Lexical File
A BASIC language program that has been processed by the interpreter into the form that is used in main memory. In this form, the program file occupies less space and loads into memory quicker. Line numbers, keywords, operators, and branch pointers are processed into binary form. Lexical files are compatible only with the version of the BASIC interpreter that generated them.

Random Access File
The contents of a random access file, such as a virtual array, can be accessed in any desired sequence by referencing individual elements in the file.

Sequential Access File
The contents of a sequential access file can only be accessed from start to end.

Transfer Rate
The rate that a file is transferred after it has been accessed.

Virtual Array
A collection of data defined by a DIMension statement that references an open channel to a file-structured device, such as the E-Disk. The data array is available to the user program just as if it were present in main memory. For this reason it is called virtual.



E-DISK ™ FUNCTIONS

John Fluke Mfg. Co., Inc.
P.O. Box 43210, Mountlake Terrace, WA 98043
800-426-0361 (toll free) in most of U.S.A.
206-774-2481 from AK, HI, WA and Canada
206-774-2398 from other countries

Fluke (Holland) B.V.
P.O. Box 5053, 5004 EB, Tilburg, The Netherlands
Tel. (013) 673973, TELEX 52237
Phone or write for the name of your local Fluke representative.

Printed in U.S.A.

B0123A-10U8011/SE EN

FLUKE®

Technical Data

1720A Highlighted Learning Program B0142

Basic Main Memory Arrays

Array variables include Reals (floating point), Integers and Strings. Interger array variable names are followed by %; string array variable names are followed by $.

Array variable names may be one or two characters (first character may be any upper case letter; second character may be any upper case letter or any number 0 through 9 or may be omitted); note these are the same names as are used with simple variables. Note also that IF, ON, OR, TO, FN, LN, PI and AS are not legal variable names.

There are only 954 possible names for each type of simple variable in FLUKE BASIC. This may seem to be adequate, however, it is not enough to store the readings from a typical digital voltmeter measurement. Moreover, the possibilities for meaningful names from the 954 possible names are very limited. Consider also, the fact that 954 BASIC statements would be required to assign a value to each possible simple variable name. For these reasons, an alternative to simple variables is necessary. The need for an array of variables (each array variable is called an element) that can be identified by a number as part of the name, is evident.

Suppose you had 10 numbers you needed to assign to simple interger variables:

```
N0% = 26792   \   N5% = 26797
N1% = 26793   \   N6% = 26798
N2% = 26794   \   N7% = 26799
N3% = 26795   \   N8% = 26800
N4% = 26796   \   N9% = 26801
```

You could do it as shown above or you could dimension an integer array of 10 elements:

DIM N%(9%) dimensions N%(0%) through N%(9%)

Each of the 10 array elements is a variable and is capable of storing data like any simple variable. In other words, DIM N%(9%) creates an additional 10 variables for the program to use.

Assignments can be made to the array element variables in the same manner as you did with the simple variables:

```
N%(0%) = 26792   \   N%(5%) = 26797
N%(1%) = 26793   \   N%(6%) = 26798
N%(2%) = 26794   \   N%(7%) = 26799
N%(3%) = 26795   \   N%(8%) = 26800
N%(4%) = 26796   \   N%(9%) = 26801
```

So far, the only difference between the simple variable assignments and the array element assignments, are the addition of brackets around the numbers, and the prerequisite DIM statement to reserve main memory space for the array variables and to establish the array element names as valid variable names in the program.

The numbers in the names of the array element variables serve the same purpose as the numbers in the simple variables, i.e., they are part of the variable name and make the named variable unique and distinguishable from the other variables. N1% is not the same variable as N%(1); both are allowed. The number within the brackets of the array element variable has one important additional capability: it can be a variable. Bracketed element numbers are usually called subscripts.

Now you can make the same assignments in a more flexible and structured manner using a FOR/NEXT loop:

```
10  DIM N%(9%)
20  FOR I% = 0% TO 9%
30  READ N%(I%)
40  NEXT I%
50  DATA 26792, 26793, 26794, 26795, 26796
60  DATA 26797, 26798, 26799, 26800, 26801
70  END
```

Enter and RUN the above program; then from the Immediate Mode, type:

```
PRINT N%(3)  <CR>
```

The display should indicate "26795" which is the 4th value in the DATA statement. The other 9 elements can be verified in like manner, or you could simply type:

```
PRINT N%(0..9)  <CR>
```

which is the equivalent of:

```
FOR I = 0 to 9
PRINT N%(I)
NEXT I
```

The above statement functions like a FOR/NEXT loop and increments the array element subscript numbers from 0 through 9; the result is to PRINT the entire array of variable elements with a <CR> <LF> after each element. To suppress the <CR><LF> type:

```
PRINT N%(0..9); <CR>
```

This notation can be used for any contiguous list of elements, e.g.,:

```
PRINT N%(3..7)
```

Now you have an even simpler method of making assignments:

```
10  DIM N%(9%)
20  READ N%(0..9)
30  DATA 26792, 26793, 26794, 26795, 26796
40  DATA 26797, 26798, 26799, 26800, 26801
50  END
```

Until now you have been working with a one-dimensional array. It may help you to think of a one-dimensional array as a one-row matrix. The above example represented as a matrix would be:

← Columns 0 through 9 →

ROW	N%(0)	N%(1)	N%(2)	N%(3)	N%(4)	N%(5)	N%(6)	N%(7)	N%(8)	N%(9)

FLUKE BASIC supports String Arrays and REAL (floating point) Arrays as well as Integer Arrays. Dimensions and Subscripts must be ≤32767 (the maximum integer size). Subscripts may appear as mathematical expressions, e.g.,:

```
V$(A% + 3)="Volts"              R(13/N)=1.1412
SN$(SQR(A%))="Serial No."       DIM N%(X% + 3%)
```

If a subscript is not an integer, BASIC rounds the subscript to an integer.

Two Dimensional (Double Subscripted) Arrays

An array element with two subscripts can be thought of as a variable with another character in its name. This additional subscript gives the program the ability to create more elements than is possible with only one subscript. The additional subscript also gives the program the capability of building a matrix with ROWs as well as COLUMNs.

Suppose you had 3 production shifts and you wanted to store the total number of instruments produced by each shift for one week (5 days). The following program is an example of one way to do this task:

```
 10  DIM PS%(2%,4%)        ! Dimension a 3x5 array
 20  READ PS%(0..2,0..4)   ! READ 15 values
 30  !   SHIFT  DAY  DAY  DAY  DAY  DAY
 40  !     1     1    2    3    4    5
 50      DATA   10,  12,   9,   7,  10
 60  !     2
 70      DATA    9,  14,  10,  11,  11
 80  !     3
 90      DATA   11,   7,   6,   8,  10
100  FOR I% = 0% TO 2%
110  PRINT PS% (I%,0..4);
120  PRINT
130  NEXT I%
140  END
```

The previous program takes the data and PRINTs it in the same arrangement as the DATA statements. Note the subscript order is always (ROW, COLUMN). The matrix for this array is shown below:

← PS$(ROW NO., 0..4) →

		Col. 1	Col. 2	Col. 3	Col. 4	Col. 5
PS%	ROW 0	PS%(0,0)	PS%(0,1)	PS%(0,2)	PS%(0,3)	PS%(0,4)
(0..2,	ROW 1	PS%(1,0)	PS%(1,1)	PS%(1,2)	PS%(1,3)	PS%(1,4)
Col. No.)	ROW 2	PS%(2,0)	PS%(2,1)	PS%(2,2)	PS%(2,3)	PS%(2,4)

Multiple Arrays

More than one array may be dimensioned in a program, e.g.,:

```
10  DIM A%(3,5), B%(6,10), A$(10,100)
20  DIM RL(1000)
30  DIM A5(10,10)
:
:
```

Redimensioning Not Allowed

BASIC programs are allowed to execute a DIM statement for each variable only once. An error will result if the program attempts to execute a specific DIM statement more than once. For this reason, DIM statements should appear early in the program and should not be included in subroutines. The only way around this is to re-RUN the program. RUN causes BASIC to forget all previously executed DIM statements.

Serial Storage of Arrays on the Disk

The following examples illustrate how array data can be serially stored and retrieved from the disk. You may use a different array name to retrieve data than you did to store the data; if the arrays are alike in type (integer, real or string) and dimensions. Note "NEW" in line 110 indicates a new file is being created on the disk.

©1982, John Fluke Mfg. Co., Inc., all rights reserved

Data Storage:

```
10 !"EXAM1.BAS"
100 CLOSE 1                              ! INITIALIZE
110 OPEN "EXAM1.DAT" AS NEW FILE 1 SIZE 1  ! RESERVE AND NAME DISK SPACE
120 DIM A$(5%)
130 FOR I%=0% TO 5%
140   A$(I%)=CHR$(65%+I%)      ! ASSIGN LETTERS "A" THROUGH "F"
150 NEXT I%
160 PRINT #1, A$(0%..5%)       ! STORE ON DISK
170 CLOSE 1                    ! CLOSE FILE
180 END
```

Data Retrieval:

```
10 !"EXAM3.BAS"
100 CLOSE 1                          ! INITIALIZE
110 OPEN "EXAM1.DAT" AS OLD FILE 1
120 DIM A$(5%)
130 !
140 INPUT LINE #1, A$(0%..5%) ! INPUT FILE DATA
150 !
160 CLOSE 1                  ! CLOSE FILE
170 PRINT A$(0%..5%)         ! DISPLAY DATA
180 END
```

Note "OLD" in line 110 indicates this file already exists on the disk.

Disk Storage Size Requirements

Before a Main Memory Array can be stored, space must be reserved for it on the disk. When a new file is OPENed, the largest available contiguous space on the storage medium (floppy disk or E-disk) is allocated for the single file, unless the SIZE is included in the OPEN statement. If two NEW files are OPENed without a SIZE statement and there is only one contiguous space available on the disk, the Operating System will display "? I/O error 306..." telling you there is no more room on the storage device, when the attempt is made to OPEN the second file. This will happen even though there is enough room on the disk for all the data you plan to store in each of the files.

Disk Size Calculation

SIZE must be stated as an integer number of BLOCKS (1 BLOCK = 512 BYTES). An Array file may contain more than one array. File SIZE must be large enough to equal or exceed the total number of storage bytes required by all of the REAL elements, INTEGER elements and STRING elements you plan to use in the Array file.

Array values are stored on the disk as ASCII values. One byte of disk storage is required for each character stored.

Disk requirements for serial storage (no comma after the variable):

1 byte per significant digit or string character
1 byte for the sign (even though it may be + and not be displayed)
1 byte for decimal point (reals only)
1 byte for an included space (except with PRINT USING) for reals and integers
2 bytes for <CR> <LF>
1 byte for EOF (end of file) character

EXAMPLES

PRINT #1, 3%	requires 5 bytes
PRINT #1, -3%	requires 5 bytes
PRINT #1, USING "S#", -3%	requires 4 bytes
PRINT #1, 3.285	requires 9 bytes
PRINT #1, -3.285	requires 9 bytes
PRINT #1, USING "S#.###", -3.285	requires 8 bytes
PRINT #1, "12345"	requires 7 bytes

Example for Disk Size Calculation

```
10  DIM S$(100), I%(100), R(100)
```

Assumptions:

1. All string elements = length of 10 characters.
2. PRINT USING is utilized to insure all reals are same length, and all integers are the same length.

```
20  CLOSE 1
30  OPEN "TEST.DAT" AS NEW FILE 1 SIZE 6
40  FOR J% = 1% TO 100%
45  S$(I%) = "1234567890" \ I%(J%)=J% \ R(J%)=J%+PI
50  PRINT #1, S$(J%)
60  PRINT #1, USING "S###", I%(J%)
70  PRINT #1, USING "S###.##", R(J%)
80  NEXT I%
90  CLOSE 1
100 END
```

	Strings	Integers	Reals	EOF
BYTE SIZE	= 100 [(10 + 2) + (4 + 2) + (7 + 2)] + 1			
	= 100 [27] + 1			
	= 2701			
BLOCK SIZE	= 2701/512 = 5.275391 } partial block not allowed; the next higher Integer = 6 Blocks			

Note: Looping 94 times instead of 100 permits a SIZE of 5 Blocks.

John Fluke Mfg. Co., Inc.
P.O. Box C9090, Everett, WA 98206
800-426-0361 (toll free) in most of U.S.A.
206-356-5400 from AK, HI, WA
206-356-5500 from other countries

Fluke (Holland) B.V.
P.O. Box 5053, 5004 EB, Tilburg, The Netherlands
Tel. (013) 673973, TELEX 52237
Phone or write for the name of your local Fluke representative.

Printed in U.S.A.
B0142A-10U8204/SE EN

FLUKE®

Technical Data

1720A Highlighted Learning Program B0143

An Introduction to Virtual Arrays

Virtual Arrays Compared to Main Memory Arrays

The 1720A BASIC provides two types of variable arrays[1] for storing and retrieving data: Main Memory (ordinary or normal) Arrays and Virtual Arrays. These two types of arrays are identical in the following ways:

Identical:

1. Array variables include Reals (floating point), Integers and Strings. Integer array names are followed by %; string array names are followed by $.
2. One or two character array variable names (first character may be any upper case letter; second character may be any upper case letter or any number 0 through 9 or may be omitted). Note that IF, ON, OR, TO, FN, LN, PI and AS are not legal variable names.
3. One and/or two dimensional arrays, i.e., one or two subscripts, enclosed in parentheses (if there are two subscripts they are separated by a comma).
4. The following BASIC statements are corrent for Main Memory Arrays and for Virtual Arrays as well:

Reals	A(1) = B3(5,J%) + Z(8)
Integers	Q2%(0) = DD%(1,1) * E%(I%)
Strings	S$(6) = "A"

   In other words, without additional program statements you cannot know which type of array (Main Memory or Virtual) the above BASIC statements represent.
5. Virtual Arrays can be assigned values from Main Memory Arrays and vice versa; Virtual Arrays can be used in equations with Main Memory Arrays.

1. Refer to Highlighted Learning Program B0142, Main Memory Arrays, for the definition and use of arrays for 1720A BASIC programming; also refer to section 2-27 and section 6 of the BASIC Programming Manual.

Virtual Arrays are different from Main Memory Arrays in the following ways:

Differences:

1. Main Memory Arrays reside in Main Memory. Virtual Arrays temporarily reside in a Main Memory Buffer of 512 bytes (1 block) per channel (file) no.; permanently reside on a storage medium (floppy disk or E-disk).
2. Main Memory Arrays are volatile because Main Memory is volatile. Virtual Arrays survive providing they have been transferred from the Main Memory Buffer to the storage medium (CLOSEing the file insures this).
3. Virtual Arrays are not initialized by the DIM statement. Main Memory Arrays are assigned initial values by the DIM statement.
4. Virtual Arrays are randomly[2] accessed from the storage medium; Main Memory Arrays must be retrieved serially[3] from the file storage medium.
5. Main Memory Arrays require PRINT # < > and INPUT # < > statements to transport data to and from the storage medium. Virtual Arrays are automatically updated on the storage medium as they are used in the program.
6. Main Memory Arrays are created with a DIM or a COM (common main memory for program chaining) statement (COM will not support string variables); Virtual Arrays are created with an OPEN and a DIM # statement.
7. Virtual Arrays can be equivalenced[4] except not with COM statements.
8. Virtual Arrays **do not** require a COM statement in order to be accessed by a chained program. COM is not needed and cannot be used with Virtual Arrays. Main Memory Arrays require a COM statement to survive program chaining.
9. Elements of Virtual Array Strings have a definite, dimensioned length. Elements of Main Memory Array strings are limited in length only by the amount of Main Memory available.

10. Since Main Memory Arrays must share main memory with the BASIC program the maximum amount of Main Memory available for Main Memory Arrays is calculated as follows (assume program occupies 5000 bytes):

    Given: Main Memory = 25,000 Bytes
    − Program Size 5,000 Bytes
    Maximum Array Size = 20,000 Bytes

    On the other hand, each Virtual Array file can be as large as 65,536 Bytes. Total number of Virtual Array files is only limited by available disk storage and the directory limit of 72 files.

11. Main Memory Arrays are stored on the disk as ASCII data; they can be viewed by the File Utility Program. Virtual Arrays are stored as Binary data and cannot be viewed by the File Utility Program.

12. Program execution errors automatically close Virtual Array files, making Virtual Array data inaccessible from the immediate mode, however, this data survives on the storage medium and can be retrieved by a program. Main Memory Arrays are accessible from the immediate mode after a program execution error.

13. Virtual Array data survives a re-RUN or EDIT of the program, but Main Memory Array data is lost in both of these situations.

2. Random Access means each variable's value is accessible without having to count through other variables which exist on the disk ahead of the desired variable.
3. Serial Access means a specific variable is accessible only by counting through the other variables which exist ahead of it on the disk.
4. Equivalenced variables share the same area of main memory.

Defining "Virtual"

After reading the above comparison of Virtual Arrays and Main Memory Arrays, it can be said that Virtual Arrays behave "virtually" the same as if they resided in Main Memory even though they actually reside on the storage medium (floppy disk or E-disk). With the exception of the OPEN, CLOSE and DIM # statements required by Virtual Arrays, the same identical programming code can be used interchangeably for Virtual Arrays or Main Memory Arrays; ignoring for the moment the fact that Virtual Array strings require some additional considerations in some situations due to their fixed length.

Virtual Array Definition

A Virtual Array is a collection of data stored in a random access file storage device, such as the electronic disk or the floppy disk. The data is stored in 1720A internal format (binary) so that no conversion is required during input or output. After a channel has been opened, the Virtual Array is available to the program just like a Main memory Array.

Advantages and Disadvantages

Virtual Arrays can be used to significantly extend the capability of a program. You will probably want to use Virtual Arrays exclusively except in situations where execution speed is critical.

Virtual Array Advantages

1. Non volatile — survives power down of 1720A; survives program chaining and DELETE ALL statements.
2. More Bytes of Main Memory are available for program storage than when Main Memory Arrays are used. Refer to the "TIME" example in the "Programming with Virtual Arrays" lesson for a typical comparison.
3. Random Access means PRINT and INPUT statements **not** necessary for data I/O.
4. Supports Equivalencing.
5. String data can be accessed by chained programs.
6. Text messages can be stored on the storage medium rather than Main Memory. Same text can be used as often as needed.
7. Program can be re-started after a 1720A power down and returned to the exact place in the program where execution ceased (due to powering down).
8. Virtual Arrays can be many times larger than Main Memory Arrays; up to 17 times (over 400K bytes) as much data can exist in Virtual Arrays when a floppy disk and two E-disks are used.

Virtual Array Disadvantages

1. Not allowed in RBYTE or WBYTE statements (see section 7 on IEEE Bus Input and Output statements, in the BASIC Programming Manual).
2. Slower execution than Main Memory Arrays. This difference in execution speeds becomes significant when large amounts of data are being sorted, assigned or operated upon. Exact speed differences are dependent on the application.
3. Unlike Main Memory Arrays where the DIM statement assigns a 0 value to Real and Integer elements and an empty string, i.e., "", to String elements (note CHR$(0) <> ""); newly created Virtual Arrays contain whatever byte arrangement that exists on the storage medium where the arrays reside.

©1982, John Fluke Mfg. Co., Inc., all rights reserved

Creating Virtual Arrays

Creating Virtual Arrays requires the following actions:

1. A filename must be associated with the Virtual Arrays.
2. A channel number must be associated with the filename.
3. A determination must be made to use NEW data (create a new disk file for new data) or OLD data (data already in a Virtual Array disk file).
4. The SIZE of the arrays in BLOCKS should be stated (continued upper right of this page).
5. The arrays must be DIMensioned.
6. The DIMension must be associated with the channel number picked in step 2, above.
7. The channel (file) must be CLOSEd in order to transport the most current array data (contained in the BUFFER) to the disk.

Main Memory Array for Comparison

The following example OPENs a file and DIMensions a Main Memory Array, then assigns values to the array and stores it on the disk. Main Memory Array data is stored on the disk in the file named "EXAM1.DAT".

```
10 !"EXAM1.BAS"
100 CLOSE 1                              ! INITIALIZE
110 OPEN "EXAM1.DAT" AS NEW FILE 1 SIZE 1  ! RESERVE AND NAME DISK SPACE
120 DIM A$(5%)
130 FOR I%=0% TO 5%
140   A$(I%)=CHR$(65%+I%)      ! ASSIGN LETTERS "A" THROUGH "F"
150 NEXT I%
160 PRINT #1, A$(0%..5%)       ! STORE ON DISK
170 CLOSE 1                    ! CLOSE FILE
180 END
```

Virtual Array

"EXAM1.BAS" has been altered (to become "EXAM2.BAS") as explained in the program comments, to use a Virtual Array instead of a Main Memory Array. Note line 140 did not change at all and line 160 is no longer needed.

Virtual Array data is stored in Virtual Array File "EXAM2.BIN".

```
10 !"EXAM2.BAS"
100 CLOSE 1                              ! INITIALIZE
110 OPEN "EXAM2.BIN" AS NEW DIM FILE 1 SIZE 1     ! added DIM and .BIN
120 DIM #1, A$(5%)                                ! added #1,
130 FOR I%=0% TO 5%
140   A$(I%)=CHR$(65%+I%)      ! ASSIGN LETTERS "A" THROUGH "F"
150 NEXT I%
160 !          not needed        PRINT #1, A$(0%..5%)      ! STORE ON DISK
170 CLOSE 1                    ! CLOSE FILE
180 END
```

Retrieving Virtual Arrays

Retrieving Virtual Arrays requires all of the seven steps mentioned in CREATING VIRTUAL ARRAYS except SIZE need not be specified. The operating system already knows the SIZE of the file named. In fact, any attempt to change the SIZE of an OLD file by including a SIZE statement in the OPEN statement, will be ignored by the Operating System. The "NEW" statement is replaced by "OLD" (if "OLD/NEW" is omitted, the program assumes "OLD").

Main Memory Array for Comparison

"EXAM1.BAS" has been altered (to become "EXAM3.BAS") to bring the data from file "EXAM1.DAT") back into Main Memory. Note line 170 is able to access the data from Main Memory even though the file has been closed. Note "NEW" has been changed to "OLD".

```
10 !"EXAM3.BAS"
100 CLOSE 1                          ! INITIALIZE
110 OPEN "EXAM1.DAT" AS OLD FILE 1
120 DIM A$(5%)
130 !
140 INPUT LINE #1, A$(0%..5%) ! INPUT FILE DATA
150 !
160 CLOSE 1                  ! CLOSE FILE
170 PRINT A$(0%..5%)         ! DISPLAY DATA
180 END
```

Virtual Array

"EXAM2.BAS" has been altered (to become "EXAM4.BAS") to retrieve the same Virtual Arrays created by "EXAM2.BAS", using the data stored in Virtual Array File "EXAM2.BIN". Note line 140 treats A$(0..5) **virtually** as if it were in Main Memory.

RUN this program and then, from the immediate mode, type: PRINT A$(0..5) <CR> and note I/O error 313 results because the Virtual Array file has been CLOSEd and A$(0..5) is **not** in Main Memory. Note "NEW" has been changed to "OLD".

```
10 !"EXAM4.BAS"
100 CLOSE 1                          ! INITIALIZE
110 OPEN "EXAM2.BIN" AS OLD DIM FILE 1
120 DIM #1, A$(5%)
130 !
140 PRINT A$(0%..5%)                 ! DISPLAY DATA
150 !
160 !
170 CLOSE 1                          ! CLOSE FILE
180 END
```

For more information on Virtual Arrays, refer to 1720A Highlighted Learning Program B0144, "Programming with Virtual Arrays"; also refer to section 6 of the 1720A BASIC Programming Manual.

John Fluke Mfg. Co., Inc.
P.O. Box C9090, Everett, WA 98206
800-426-0361 (toll free) in most of U.S.A.
206-356-5400 from AK, HI, WA
206-356-5500 from other countries

Fluke (Holland) B.V.
P.O. Box 5053, 5004 EB, Tilburg, The Netherlands
Tel. (013) 673973, TELEX 52237
Phone or write for the name of your local Fluke representative.

Printed in U.S.A.

B0143A-10U8204/SE EN

FLUKE®

Technical Data

1720A Highlighted Learning Program B0144
Programming With Virtual Arrays

Routines No Virtual Array Should Be Without

Although Virtual Arrays behave "virtually" the same as Main Memory Arrays, these arrays do require some special handling to prevent unexpected events such as: loss of data, data not stored on disk, disks that become filled without your knowledge, tabs that don't work, disk garbage appearing as initial array values and strings that won't concatenate because they are filled with "null" characters that you can't see.

The program on the following page illustrates the use of "handlers" which should always be used with Virtual Arrays to prevent these unexpected events.

The purpose of the program will now be explained:

Escape Sequences (lines 100 through 160)

This module provides display enhancements and has no direct effect on the Virtual Array.

Open Disk File to Contain Virtual Arrays
(lines 1000 through 1130)

This module creates the Virtual Array and provides several protections against unexpected results. The "ON CTRL/C" handler (line 1020) guarantees the disk file will always be CLOSEd if the user interrupts the program. This file must be CLOSEd when the program is interrupted in order to:

1. Copy the contents of the buffer onto the disk, insuring that the data won't be lost, and,
2. Delete the previous Virtual Array disk file by having the temporary disk file take its place, insuring no data is lost and that an extra file does not remain on the disk.

Two OPEN statements (lines 1050 and 1070) are used. Line 1050 assumes a data file already exists out on the disk and attempts to OPEN this file. If no file is found, an error is generated and the ON ERROR branch (line 1030) takes over at line 1070 and a NEW file is created, otherwise the OLD file is OPENed. If a NEW file is created, the flag FG% is set (line 1080) to insure the NEW arrays get meaningful initial values (line 1120).

```
10  ! "EXAM5"
20  !
30  ! THIS PROGRAM DEMONSTRATES PROPER PROGRAMMING TECHNIQUES
40  ! FOR A VIRTUAL ARRAY.  THE PROGRAM ALLOWS THE USER TO
50  ! INPUT OR KEEP SIX NAMES OF NO MORE THAN 32 CHARACTERS.
60  ! CHARACTER LENGTHS OF NAMES ARE DISPLAYED.
70  !
100 !**************** ESCAPE SEQUENCES *****************
110 !
120 ES$=CHR$(27%)+"["          ! ESCAPE
130 CH$=ES$+"2J"+ES$+";H"      ! CLEAR SCREEN AND HOME CURSOR
140 EL$=ES$+"K"                ! ERASE TO END OF LINE
150 SL$=ES$+"1K"               ! ERASE TO START OF LINE
160 !
1000 !****** OPEN DISK FILE TO CONTAIN VIRTUAL ARRAYS ******
1010 !
1020 ON CTRL/C GOTO 2150       ! CLOSE FILES BEFORE EXITING PROGRAM
1030 ON ERROR GOTO 1070        ! BRANCH WHEN "NAMES.BIN" DOESN'T EXIST
1035 CLOSE 1                   ! VERIFY CHANNEL 1 HAS NOT BEEN LEFT OPEN
1040 FG%=0                     ! FLAG TO INDICATE NEW DATA
1050 OPEN "NAMES.BIN" AS OLD DIM FILE 1
1060 GOTO 1100                 ! JUMP AROUND "NEW" OPEN STATEMENT
1070 OPEN "NAMES.BIN" AS NEW DIM FILE 1 SIZE 1
1080 FG%=1%                    ! INDICATE "NEW" DATA
1090 RESUME 1100               ! EXIT FROM "ON ERROR" BRANCH
1100 OFF ERROR                 ! CURRENT ERROR HANDLER NO LONGER NEEDED
1110 DIM #1, NM$(5%)=32%       ! CREATE 6 STRING ELEMENTS OF 32 CHARACTERS EACH
1120 IF FG% THEN GOSUB 20020 \ FG%=0   ! INITIALIZE NEW DATA
1130 !
```

Program Listing is continued on next page

```
2000 !****************** ENTER NAMES ********************
2010 !
2020 ON ERROR GOTO 30020          ! HANDLE INPUT ERRORS
2030 PRINT CH$                    ! CLEAR SCREEN AND HOME CURSOR
2040 FOR I%=0% TO 5%
2050 PRINT CPOS(2,15);"NAME NO.";I%;"IS CURRENTLY ";EL$;NM$(I%)
2060 VA$=NM$(I%)                  ! PASS TO MAIN MEMORY VARIABLE FOR SUBROUTINE
2070 GOSUB 21020                  ! STRIP OFF NULL CHARACTERS
2080 PRINT CPOS(3,15);"STRING LENGTH =";LEN(VA$),"LEN(NM$(I%)=";LEN(NM$(I%))
2090 PRINT CPOS(7,15);"ENTER NAME <CR>"
2100 PRINT CPOS(5,15);EL$;        ! POSITION CURSOR AND ERASE PREVIOUS ENTRY
2110 INPUT LINE T$
2120 IF T$="" GOTO 2140           ! OPTION TO KEEP PREVIOUS ENTRY
2130 NM$(I%)=T$                   ! TRANSFER ENTRY FROM TEMPORARY STORAGE
2140 NEXT I%
2150 CLOSE 1                      ! COPY BUFFER CONTENTS ONTO DISK
2160 END
2170 !
20000 !***** ASSIGN MEANINGFUL INITIAL VALUES IN LIEU OF DISK GARBAGE *****
20010 !
20020 FOR J%=0% TO 5%
20030   NM$(J%)="YOUR CHOICE"
20040 NEXT J%
20050 RETURN
20060 !
21000 !**************** DELETE NULL CHARACTERS ***************
21010 !
21020 P1%=INSTR(1%,VA$,CHR$(0%))          ! LOCATE 1ST NULL POSITION
21030 SG%=SGN(P1%)                        ! WAS NULL FOUND?
21040 P2%=P1%-1%                          ! LOCATE LAST NON-NULL CHARACTER
21050 VA%=(1%-SG%)*LEN(VA$)+P2%*SG%       ! LENGTH OF CHARACTER STRING W/O NULLS
21060 VA$=LEFT(VA$,VA%)                   ! STRIP OFF NULL CHARACTERS
21070 RETURN
21080 !
30000 !************ ERROR HANDLER FOR INPUT ERRORS **********
30010 !
30020 PRINT CHR$(7%);CPOS(9,15);"ILLEGAL ENTRY";
30030 WAIT 1000
30040 PRINT SL$;     ! ERASE "MESSAGE"
30050 RESUME 2100
30060 !
32030 END
```

Enter Names (lines 2000 through 2170)

This module provides the display prompts and INPUT LINE statement to allow the user to enter six names. Notice the use of the temporary variable T$ (lines 2110 and 2120). If the user makes no entry before pressing the RETURN key, the BASIC System will input an empty string to T$ and the data contained in NM$ (I%) will remain the same.

The length of NM$(I%) and the relevant string length is displayed (line 2080) to demonstrate the need to strip off the Null Characters (line 2070). Any attempt to concatenate NM$(I%), e.g., NM$(I%) = NM$(I%) + "anything" will result in an error because Virtual Array string elements are always full due to Null Characters which are added by the BASIC System. Proper concatenation could be accomplished as follows:

```
2060 VA$ = NM$(I%)          ! Virtual Array Assigned to
                              Main Memory Array
2070 GOSUB 21020            ! Strip Off Null Characters[5]
2172 VA$ = VA$ + "anything" ! Final LEN (VA$) ≤32
2174 NM$ (I%) = VA$         ! Assignment to Virtual Array
```

[5] This could also be done with a DEF FN (see section 6-22 of the BASIC programming manual). Although somewhat slower than a subroutine, the defined function eliminates the need for dedicated variable names.

Assign Meaningful Initial Values in Lieu of Disk Garbage (lines 20000 through 20060)

This subroutine is only used if a NEW data file is OPENed. For an interesting experiment, type: KILL "NAMES.BIN" <CR> from the immediate mode. Then go to FUP and format a new disk and copy your original disk onto the new disk. Using the new disk, get into BASIC and load (OLD) the program. Then delete line 1120. RUN the program and observe the disk garbage. Had Real or Integer arrays been used this module could have assigned zeros to these array elements.

©1982, John Fluke Mfg. Co., Inc. All rights reserved. Litho in U.S.A.

Error Handler for Input Errors

(lines 30000 through 30060)

This subroutine prevents program interruption by handling illegal data entries. RUN the program and make entries which are longer than 32 characters and watch this error handler in action. The branch to this error handler is set-up in line 2020 of the program.

User Information

This program may be interrupted at any time by a CTRL C, and all entries made just prior to the CTRL C will be stored on the disk. In other words, if all entries are correct except the 3rd name, RUN the program and simply press RETURN until the 3rd name is displayed; enter the 3rd name, press RETURN, then CTRL C and you are finished.

Debugging Programs That Use Virtual Arrays

Most programmers will attempt to PRINT variable values from the Immediate Mode after their program has been interrupted by an error. The variable values can give clues as to what went wrong with the program. It is **not** possible to PRINT Virtual Array values from the Immediate Mode after the program has been interrupted by an error, because the BASIC System reacts to the error by CLOSEing all Virtual Array files in order to protect their data. The same problem exists for a CTRL C interrupt when the program contains an ON CTRL/C branch to CLOSE the Virtual Array Files.

Here are two ways to overcome this problem:

1. Use an ON ERROR handler which assigns key Virtual Array values to Main Memory variables and then CLOSEs the Virtual Array files

   or

2. Develop the program using Main Memory Arrays. If available Main Memory is lacking, develop a module at a time using Main Memory Arrays. In the previous program all that would have to happen to convert it to Main Memory Arrays is:

   Delete or Comment out (!) lines: 1030 through 1100, 1120 and 2150 Change line 1110 to "DIM NM$(5%)"

After the program has been debugged, add the original versions of the above lines back into the program and take advantage of the Virtual Arrays, once again.

Single Dimension to Double Dimension Equivalencing

The following example shows how to equivalence a single dimension array with a two dimension array having the same number of elements. Delete the "%" from the variable names and you will see it works for real variables as well. It also works for strings.

```
10 ! "EXAM6A" USES INTEGER-INTEGER EQUIVALENCING
20 !          TO EQUIVALENCE ONE AND TWO DIMENSION ARRAYS
30 CLOSE 1
40 OPEN "DATA.BIN" AS NEW DIM FILE 1 SIZE 1
50 DIM #1, A2%(1%,3%)
60 DIM #1, A1%(7%)
70 FOR I%=0% TO 7%
80   READ A1%(I%)     ! READING FOR THE A1%(ARRAY) DOES IT FOR THE A2%(ARRAY)
90 NEXT I%
100 PRINT "DISPLAY ONE-DIM ARRAY"
110 PRINT
120 FOR I%=0% TO 7% \ PRINT A1%(I%);"  "; \ NEXT I%
130 PRINT \ PRINT
140 PRINT "DISPLAY TWO-DIM EQUIVALENCED ARRAY"
150 PRINT
160 FOR J%=0% TO 3%
170   PRINT A2%(0%,J%), A2%(1%,J%)
180 NEXT J%
190 CLOSE 1
200 DATA 100, 200, 300, 400, 500, 600, 700, 800
210 END
```

Integer-String Equivalencing

The following example shows how to create the 16 character string "JOHN FLUKE MFG.?" using 8 integer values. The Virtual String Array A$(0) and the Virtual Integer Array share the same disk space. To create your own message, first assign your message to A$(0), i.e., A$(0) = "Your Message," then PRINT A%(0%..7%) to see what integer values are required.

```
10 ! "EXAM6" USES INTEGER-STRING EQUIVALENCING
20 !          TO SPELL "JOHN FLUKE MFG.?"
30 CLOSE 1
40 OPEN "DATA.BIN" AS NEW DIM FILE 1 SIZE 1
50 DIM #1, A$(0)=16
60 DIM #1, A%(7%)
70 FOR I%=0% TO 7%
80   READ A%(I%)
90 NEXT I%
100 DATA 20298,20040,17952,21836,17739,19744,18246,16174
110 PRINT A$(0)
120 CLOSE 1
130 END
```

Result: "JOHN FLUKE MFG.?" is displayed.

The next example shows how to assign the integer values, 1 through 8 to the Virtual Integer Array A%(0%..7%) using a 16 character Virtual String Array A$(0). These two virtual arrays share the same disk space. To create your own integer numbers, first assign these numbers to A%(0%..7%) then look at each of the 16 characters and determine its ASCII value, e.g.,

```
FOR I% = 1% TO 16%
PRINT ASCII (MID (A$(0), I%, 1%))
NEXT I%
```

Place these ASCII values in the DATA statement on line 160.

```
10 ! "EXAM7" USES INTEGER-STRING EQUIVALENCING
20 !          TO ASSIGN A%(0%)=1,A%(1%)=2,...,A%(7%)=8
30 CLOSE 1
40 OPEN "DATA2.BIN" AS NEW DIM FILE 1 SIZE 1
50 DIM #1, A$(0)=16
60 DIM #1, A%(7%)
70 T$=""
80 FOR I%=1% TO 16%
90   READ N%
100  T$=T$+CHR$(N%)
110 NEXT I%
120 A$(0)=T$
130 PRINT "A%(0%..7%) =";A%(0%..7%);
140 PRINT \ PRINT
150 PRINT "A$(0%) = ";A$(0%)
160 DATA 1,0,2,0,3,0,4,0,5,0,6,0,7,0,8,0
170 CLOSE 1
180 END
```

Result: "A%(0%..7%) = 1 2 3 4 5 6 7 8" is displayed.
"A$(0%) = ┌┴┘≒☒✓ " is displayed.

Creating Dimension Statements Requiring More Than One Program Line

Sometimes one line won't hold all the Virtual Arrays you would like to DIMension. In the examples below the DIM statement started on line 70 is continued on line 80 by equivalencing a dummy array. The dummy array occupies the same memory space as all the arrays on line 70. The dummy array must contain the total number of bytes for all the arrays on line 70 plus the number of bytes, if any, that are left vacant prior to overlapped Block boundaries.

```
10 ! "EXAM8"
20 ! EQUIVALENCING TO ADD TO THE DIMENSION STATEMENT.
30 ! FF%(ARRAY) IS A DUMMY ARRAY EQUIVALENCED TO THE ARRAYS FOR THE
40 ! DIM STATEMENT ON LINE 70
50 CLOSE 1
60 OPEN "DATA3.BIN" AS NEW DIM FILE 1 SIZE 5
70 DIM #1, AA$(10%)=32%,BB$(10%)=32%,CC$(10%)=32%,DD$(10%)=32%,EE$(10%)=32%
80 DIM #1, FF%(5%*11%*32%/2),FF$(10%)=32%
90 CLOSE 1
100 END
```

```
10 ! "EXAM8A"
20 ! EQUIVALENCING TO ADD TO THE DIMENSION STATEMENT.
30 ! HH%(ARRAY) IS A DUMMY ARRAY EQUIVALENCED TO THE ARRAYS FOR THE
40 ! DIM STATEMENT ON LINE 70
50 CLOSE 1
60 OPEN "DATA3.BIN" AS NEW DIM FILE 1 SIZE 3
70 DIM #1, AA$(10%)=32%,BB$(10%)=32%,CC(10%),DD(10%),EE(10%),FF%(10%),GG%(10%)
80 DIM #1, HH%(11%*(32+32+8+8+8+2+2)/2),II%(10%)
90 CLOSE 1
100 END
```

Using the Tab Function With Virtual Array Strings

The Null Characters in Virtual Array Strings cause unexpected events in PRINT and PRINT USING statements which also include the TAB function. In order to avoid this, assign the Virtual Array String element to a Main Memory String variable, strip out the Null Characters from this Main Memory variable then use it in the PRINT or PRINT USING statement in place of the Virtual Array element. Example 9 compares TAB printing with and without Null Characters.

```
10 ! "EXAM9"
20 ! VIRTUAL ARRAYS WITH THE TAB FUNCTION
30 CLOSE 1,2
40 OPEN "KB0:" AS NEW FILE 2
50 OPEN "DATA.VRT" AS NEW DIM FILE 1 SIZE 1
60 !
70 !        COLUMN REFERENCE
80 PRINT "0        1         2         3"
90 PRINT "12345678901234567890123456789O"
100 !
110 DIM #1, C$(0%)
120 C$(0%)="1234567890"     ! ASSIGN 10 CHARACTERS TO 16 CHARACTER ELEMENT
130 VA$=C$(0%)              ! SUBROUTINE PASS VARIABLE
140 GOSUB 220               ! FIND LENGTH OF STRING W/O NULLS
150 C$=LEFT(C$(0%),VA%)     ! STRIP NULLS
160 PRINT #2,"";TAB(7%);C$;TAB(27%);PI          ! PRINT WITHOUT NULLS
170 PRINT #2
180 PRINT #2,"";TAB(7%);C$(0%);TAB(27%);PI      ! PRINT WITH NULLS
190 CLOSE 1,2
200 END
```

Program Listing is continued on next page

```
210 !************** LENGTH OF STRING W/O NULLS *****************
220 !
230 P1%=INSTR(1%,VA$,CHR$(0%))
240 SG%=SGN(P1%)
250 P2%=P1%-1%
260 VA%=(1%-SG%)*LEN(VA$)+P2%*SG%
270 RETURN
280 END
```

How to Delete <not used> and <temp ent> Files

<not used> and <temp ent> files accumulate on the disk when Virtual Array files are OPENed and not CLOSEd prior to terminating or EDITing a program. These files can only be seen by the File Utility Program (FUP) using the /E command, but they do take up disk memory and can quickly fill a disk. These files can only be deleted by packing the disk (/P command). Note: These files are **not** transferred using the /W command.

Programming For Faster Execution

The following programming techniques can make a significant difference in the time it takes to execute a program.

Open More Channels

A buffer in Main Memory can only contain 1 Block (512 bytes) of a Virtual Array. When your program uses an array element that is stored in a different block than the block in the buffer, the BASIC System must take the block from the buffer and store it on the disk, then copy the other block into the buffer. Every time this happens, the program must wait until the block transfers are complete, resulting in a loss of program execution speed.

Obviously, if we could get more than one block of the Virtual Arrays into Main Memory Buffer, fewer block transfers would be required during program execution, with a resulting increase in execution speed. To accomplish this, split the Virtual Arrays up into as many as 6 separate groups; OPEN a channel number (1 through 6) for each group. This establishes a one block buffer in Main Memory for each channel OPENed. You will, of course, need a separate file name for each channel OPENed. For example:

In place of —

OPEN "DATA.BIN" AS NEW FILE 1 SIZE 11

DIM #1, A (100%), B (100%), C (100%), D (100%), E (100%), F (100%)

use —

OPEN "DATAA" AS NEW FILE 1 SIZE 2
OPEN "DATAB" AS NEW FILE 2 SIZE 2
OPEN "DATAC" AS NEW FILE 3 SIZE 2
OPEN "DATAD" AS NEW FILE 4 SIZE 2
OPEN "DATAE" AS NEW FILE 5 SIZE 2
OPEN "DATAF" AS NEW FILE 6 SIZE 2

DIM#1, A (100%)
DIM#2, B (100%)
DIM#3, C (100%)
DIM#4, D (100%)
DIM#5, E (100%)
DIM#6, F (100%)

Efficient Element Access For Two-Dimensional Arrays

The fastest assignment of values to a large portion of a double subscripted Virtual Array occurs when the program associates the outside loop with the leftmost subscript; and associates the inside loop with the rightmost subscript as shown in the example "TIME" below:

```
4000 ! "TIME"
4010 CLOSE1
4020 OPEN "MF0:TEMP.DAT" AS NEW DIM FILE 1 SIZE 16
4030 DIM #1, A%(63%,63)
4040 T1=TIME
4050 FOR I%=0% TO 63%
4060   FOR J%=0% TO 63%
4070     A%(I%,J%) = 0%
4080   NEXT J%
4090 NEXT I%
```

Program Listing is continued on next page

```
4100 T2=TIME
4110 PRINT "UNUSED MEMORY = ";MEM
4120 CLOSE 1
4130 PRINT CHR$(7%)
4140 KILL"MF0:TEMP.DAT"
4150 PRINT "TOTAL TIME = ";(T2-T1)/1000;" SEC"
4160 END
```

Refer to the TABLE below for a comparison of execution speeds for both subscript arrangements.

Take Advantage of the Electronic Disk

Storing Virtual Arrays on the E-Disk can make them execute almost as fast as Main Memory Arrays. In the example "TIME" above, change "MF0:" to "ED0:" in lines 4020 and 4140 and note the increase in execution speed.

Note that the Main Memory Arrays in the example "TIME" require 7604 more bytes of Main Memory than the Virtual Arrays.

TABLE OF EXECUTION SPEED AND UNUSED MAIN MEMORY FOR EXAMPLE "TIME"

	Floppy Disk	E-Disk	Main Memory
Program Lines	Use "MF0:" in lines 4020 and 4140	Use "ED0:" in lines 4020 and 4140	Delete 4010, 4020, 4120, 4140; 4030 reads DIM A% (63%,63%)
4070 A%(I%,J%) = 0%	29.12 sec	22.36 sec	21.03 sec
4070 A%(J%,I%) = 0%	461.3 sec	31.53 sec	21.03 sec
Unused Main Memory (MEM)	24274 Bytes	24274 Bytes	16670 Bytes

Use the Same DIM Statement for More Than One Open Statement

If you have groups of data which are identical as in the following example, one DIM # statement is all that is necessary. An error will result if you attempt to execute a second DIM # statement for the same channel no.

```
10 ! "EXAM10"
20 ! MULTIPLE USE OF A SINGLE DIM STATEMENT
30 !
40 CLOSE 1
50 OPEN "DATA4.BIN" AS NEW DIM FILE 1 SIZE 1
60 DIM #1, A$(3%)=64%
70 A$(0%)=CPOS(4,20)+"TEXT CAN BE STORED IN VIRTUAL"
80 A$(1%)=CPOS(5,20)+"ARRAYS FOR USE AS A PROMPT AS"
90 A$(2%)=CPOS(6,20)+"MANY TIMES AS IT IS NEEDED BY"
100 A$(3%)=CPOS(7,20)+"BY YOUR PROGRAM"
110 PRINT A$(0%..3%)
120 CLOSE 1
130 OPEN "DATA5.BIN" AS NEW DIM FILE 1 SIZE 1
150 A$(0%)=CPOS(10,20)+"BY USING THE SAME PROMPT MANY"
160 A$(1%)=CPOS(11,20)+"TIMES, BUT ONLY HAVING TO CODE"
170 A$(2%)=CPOS(12,20)+"IT ONCE AS A VIRTUAL ARRAY, WILL"
180 A$(3%)=CPOS(13,20)+"REQUIRE MUCH LESS MAIN MEMORY"
190 PRINT A$(0%..3%)
200 CLOSE 1
210 END
```

Chaining Programs Which Use Virtual Arrays

Virtual Arrays can be used in lieu of the COM statement, in fact, Virtual Arrays are better because they allow string arrays to survive; COM statements do not. The program "EXAM11", below, is meant to be chained by the program "CHAIN". "EXAM10" was altered as follows to accomplish the chaining:

insert line 205 RUN "EXAM11"

delete lines 110 and 190 in "EXAM10" as they are no longer necessary

```
10 ! "CHAIN"
20 ! MULTIPLE USE OF A SINGLE DIM STATEMENT
30 !
40 CLOSE 1
50 OPEN "DATA4.BIN" AS NEW DIM FILE 1 SIZE 1
60 DIM #1, A$(3%)=64%
70 A$(0%)=CPOS(4,20)+"TEXT CAN BE STORED IN VIRTUAL"
80 A$(1%)=CPOS(5,20)+"ARRAYS FOR USE AS A PROMPT AS"
90 A$(2%)=CPOS(6,20)+"MANY TIMES AS IT IS NEEDED BY"
100 A$(3%)=CPOS(7,20)+"BY YOUR PROGRAM"
120 CLOSE 1
130 OPEN "DATA5.BIN" AS NEW DIM FILE 1 SIZE 1
150 A$(0%)=CPOS(10,20)+"BY USING THE SAME PROMPT MANY"
160 A$(1%)=CPOS(11,20)+"TIMES, BUT ONLY HAVING TO CODE"
170 A$(2%)=CPOS(12,20)+"IT ONCE AS A VIRTUAL ARRAY, WILL"
180 A$(3%)=CPOS(13,20)+"REQUIRE MUCH LESS MAIN MEMORY"
200 CLOSE 1
205 RUN "EXAM11"
210 END
```

```
10 ! "EXAM11"
20 ! CHAINING PROGRAMS WHICH USE EXISTING VIRTUAL ARRAY DATA
30 ! USE "EXAM10" TO CHAIN THIS PROGRAM IN
40 CLOSE 1
50 OPEN "DATA4.BIN" AS OLD DIM FILE 1
60 DIM #1, A$(3%)=64%
110 PRINT A$(0%..3%)
120 CLOSE 1
130 OPEN "DATA5.BIN" AS OLD DIM FILE 1
190 PRINT A$(0%..3%)
200 CLOSE 1
210 END
```

Additional information on program chaining with Virtual Arrays is found in section 10 of the FLUKE BASIC Programming Manual. This section includes an excellent example on how to re-start a specified program after a power interruption using Virtual Arrays.

NOTE: It is not necessary to CLOSE and re-OPEN channels when chaining in a program to the same Virtual Arrays, however, to prevent unexpected results due to errors, it is recommended that the CLOSEing and re-OPENing always be done.

Using Partial Arrays

If you only want the leading portion of the data in a Virtual Array, it is possible to DIMension for just that portion your program requires. "EXAM11" has been altered to only use the first element of the file without disturbing the rest of the arrays in the file. This appears on the next page as "EXAM12".

```
10 ! "EXAM12"
20 ! USING PORTIONS OF VIRTUAL ARRAYS
30 ! USES "EXAM10" VIRTUAL ARRAY DATA
40 CLOSE 1
50 OPEN "DATA4.BIN" AS OLD DIM FILE 1
60 DIM #1, A$(0%)=64%
110 PRINT A$(0%)
120 CLOSE 1
130 OPEN "DATA5.BIN" AS OLD DIM FILE 1
190 PRINT A$(0%)
200 CLOSE 1
210 END
```

Dimensioning for Disk Storage Space Economies

Section 6-24 through 6-33 of the FLUKE BASIC Programming Manual and the Help Lesson "An Introduction to Virtual Arrays" describe why the order that Virtual Array declarations appear in the DIM statement, affects the amount of disk storage required.

The rule of allocating virtual array declarations from left to right in decreasing order of array element lengths is all that is needed to insure disk storage space efficiency, except when a different declaration order is necessary to facilitate tasks such as accessing the leading portion of an array file (see "EXAM12" program, above). The program "EXAM14", below, computes the disk storage space required for all possible orders of DIM statement declaration.

If you need help in visualizing how the bytes per element are allocated for storage on the disk, study the algorythm in "EXAM14" (lines 1160 through 1340). "EXAM14" calculates the most efficient DIMensioning for a group of three Virtual Arrays (see line 750). This program prints out all possible combinations; indicating the disk storage space required for each combination. The program allocates bytes and blocks just as the BASIC System would do it, in the order that the arrays appear in the DIM statement. The printout appears on the 1720A display, however, the printout is easily changed to an RS-232-C port by changing "KB0:" to "KB1:" or "KB2:" in line 450.

"EXAM14" can easily be altered to try combinations for more or less than three arrays. For example, suppose a 4th array "F$(20%,2%)=8%" were added to the existing DIM statement on line 450; these program lines would be updated or added as indicated:

```
520  N% = 4%
525  NF% = 4% * 3% * 2% * 1%
740  add     " ... 4 (order: 1234)"
750  add     " ... ,F$ (20%, 2%) = 8%"
```

Note: Delete comment on line 750 to make room.

```
add:  822    FOR D% = 1% TO N% ! D% = position
             of array no. 4
add:  823    IF D% = A% OR D% = B% OR D% =
             C% GOTO 2165
      825    add \ AP$(D%) = "4"! delete comment
             to make room
      840    add ... + AP$ (4%)
add:  935    NE%(D%) = 63% \ NY%(D%) = N3% !
             Comment
      950    delete "THREE" insert "FOUR"
      960    update for FOUR FACTORIAL (4 x 3
             x 2 x 1 = 24 Combinations)
add:  2165   NEXT D%
```

After RUNning the program for the fourth array, the results for best choice should be: 3240 or 3421.

```
10 ! EXAMPLE 14
20 ! VERSION 1.0, 30 DECEMBER 1981
25 !
26 ! THIS PROGRAM CALCULATES THE SEQUENCE OF VIRTUAL ARRAY DIMENSIONING
27 ! WHICH WILL REQUIRE THE LEAST AMOUNT OF DISK STORAGE
28 !
30 PRINT CHR$(27%)+"[2J"   ! CLEAR DISPLAY
40 !
50 L$="S###.##"            ! PRINT USING FORMAT FOR BLOCKS
60 Y$="S####"              ! PRINT USING FORMAT FOR BYTES
70 !
100 ! DEFINITION OF VARIABLES
110 !
```

Program Listing is continued on next page

```
120 ! (Y) STANDS FOR B(Y)TE IN THE VARIABLE NAMES
130 ! (L) STANDS FOR B(L)OCK IN THE VARIABLE NAMES
140 !
150 ! A% = ORDER SUBSCRIPT
160 ! B% = ORDER SUBSCRIPT
170 ! C% = ORDER SUBSCRIPT
180 ! K% = COUNTER FOR THE SIX POSSIBLE SEQUENCES
190 ! I% = ARRAY NO.
200 ! J% = ELEMENT COUNTER
210 ! L% = BLOCK COUNTER
220 ! N% = TOTAL NO. OF ARRAYS TO BE DIMENSIONED
230 ! Y% = BYTE COUNTER
240 !
250 ! AY% = ACTUAL TOTAL BYTES USED BY ALL THE ARRAYS
260 ! TY% = MINIMUM TOTAL BYTES REQUIRED FOR ALL THE ARRAYS
270 ! AL  = ACTUAL TOTAL BLOCKS USED BY ALL THE ARRAYS
280 ! TL  = MINIMUM TOTAL BLOCKS REQUIRED FOR ALL THE ARRAYS
290 ! MN% = MINIMUM UNUSED BYTES
300 !
310 ! EL%(I%) = ENDING BLOCK NO. FOR ARRAY I%
320 ! EY%(I%) = ENDING BYTE NO. WITHIN BLOCK NO. EL%(I%) FOR ARRAY I%
330 ! NE%(I%) = NO. OF ELEMENTS IN ARRAY I%
340 ! NY%(I%) = NO. OF BYTES PER ELEMENT IN ARRAY I%
350 ! SL%(I%) = STARTING BLOCK NO. FOR ARRAY I%
360 ! SQ$(K%) = SEQUENCE OF DIMENSIONING
370 ! SY%(I%) = STARTING BYTE NO. WITHIN BLOCK NO. SL%(I%) FOR ARRAY I%
380 ! SZ%(I%) = TOTAL NO. OF BYTES REQUIRED BY ARRAY I%
390 ! UB%(K%) = UNUSED BYTES FOR THE KTH SEQUENCE TEST
400 ! UY%(I%) = ACCUMULATOR OF NO. OF UNUSED BYTES IN BLOCKS USED BY ARRAY I%
410 ! AP$(1..N%) = ARRAY POSITION
415 !
420 !***************** PICK OUTPUT DEVICE ***************
430 !
440 CLOSE 1
450 OPEN "KBO:" AS NEW FILE 1
460 !
500 !***************** INITIALIZATION ***************
510 !
520 N%=3%       ! NO. OF ARRAYS TO BE DIMENSIONED
525 NF%=3%*2*1% ! = N% FACTORIAL POSSIBLE DIMENSIONING COMBINATIONS
530 RL%=8%      ! NO. OF BYTES REQUIRED BY A REAL NUMBER
540 NT%=2%      ! NO. OF BYTES REQUIRED BY AN INTEGER
550 N1%=2%      ! STRING LENGTH OF 2 CHARACTERS
570 N2%=4%      !                  4
575 N3%=8%      !                  8
580 N4%=16%     !                 16
590 N5%=32%     !                 32
600 N6%=64%     !                 64
610 N7%=128%    !                128
620 N8%=256%    !                256
630 N9%=512%    !                512
640 !
650 DIM SY%(N%),SL%(N%),EY%(N%),EL%(N%),UY%(N%),NE%(N%),NY%(N%),SZ%(N%)
660 DIM SQ$(NF%), UB%(NF%), AP$(N%)
670 K%=0
680 !
700 !********* ASSIGN SUBSCRIPTS FOR THE SIX POSSIBLE SEQUENCES ******
710 !
720 T1$="          SUBSCRIPTS WHICH IDENTIFY ARRAYS"
730 T2$="          ------------------------------"
740 T3$="ARRAY NO.    1          2          3       (ORDER:  123)"
750 T4$=" DIM #1, G%(1%,4%), E(10%,9%), Q$(10%)=64%   ! ARRAYS BEING TESTED"
760 PRINT #1, T1$ \ PRINT #1, T2$ \ PRINT #1, T3$ \ PRINT #1, T4$ \ PRINT #1
770 !
780 FOR A%=1% TO N%            ! A% = POSITION OF ARRAY NO. 1
790 FOR B%=1% TO N%            ! B% = POSITION OF ARRAY NO. 2
800   IF B%=A% GOTO 2180
810 FOR C%=1% TO N%            ! C% = POSITION OF ARRAY NO. 3
820   IF C%=A% OR C%=B% GOTO 2170
825     AP$(A%)="1" \ AP$(B%)="2" \ AP$(C%)="3"  ! ARRAY POSITION STRINGS
830     K%=K%+1%                  ! INCREMENT TEST COUNTER (1% TO NF%)
840     SQ$(K%)=AP$(1%)+AP$(2%)+AP$(3%)  ! STORE ACTUAL SEQUENCE
850 !
```

Program Listing is continued on next page

```
860 ! SUBSCRIPTS IN THE SIX VARIABLES BELOW CORRESPOND
870 ! WITH THE SELECTED DIMENSIONED ORDER BEING TESTED
880 !
890 ! NO. OF ELEMENTS -- BYTES PER ELEMENT -- ORDER --          --  ARRAY NO.
900 !
910     NE%(A%)=10%     \     NY%(A%)=NT%    ! A%TH ARRAY TO BE DIMENSIONED (1)
920     NE%(B%)=110%    \     NY%(B%)=RL%    ! B%TH ARRAY TO BE DIMENSIONED (2)
930     NE%(C%)=11%     \     NY%(C%)=N6%    ! C%TH ARRAY TO BE DIMENSIONED (3)
940 !
950 ! NOTE THERE ARE THREE FACTORIAL POSSIBLE DIMENSION COMBINATIONS
960 ! FOR THE ABOVE THREE ARRAYS: 123, 132, 213, 231, 312, 321
970 !
1000 !*************** MAIN PROGRAM ************************
1010 !
1020 Y%=0           ! INITIALIZE BYTE COUNTER
1030 L%=1%          ! INITIALIZE BLOCK COUNTER
1040 UB%(K%)=0      ! INITIALIZE TOTAL UNUSED BYTE ACCUMULATOR
1050 !
1060 !                    ----- DISPLAY HEADINGS -----
1070 !
1080 PRINT #1,TAB(34);"ORDER: ";SQ$(K%)
1090 PRINT #1,TAB(9);"ARRAY";TAB(16);"STARTING";TAB(29);"STARTING";
1100 PRINT #1,TAB(44);"ENDING";TAB(57);"ENDING";TAB(70);"UNUSED"
1110 PRINT #1,TAB(9);"NO.";TAB(18);"BYTE";TAB(30);"BLOCK";
1120 PRINT #1,TAB(45);"BYTE";TAB(57);"BLOCK";TAB(70);"BYTES"
1130 !
1140 !               ---- ALLOCATE STORAGE TO BLOCKS ----
1150 !
1160 FOR I%=1% TO N%                ! ARRAY NO. COUNTER
1170   SZ%(I%)=NE%(I%)*NY%(I%)      ! TOTAL BYTES REQUIRED BY THE ARRAY
1180   SY%(I%)=Y%+1%          ! STARTING BYTE NO.
1190   SL%(I%)=L%             ! STARTING BLOCK NO.
1200   UY%(I%)=0              ! INITIALIZE UNUSED BYTE ACCUMULATOR
1210 !
1220       FOR J%=1% TO SZ%(I%) STEP NY%(I%)   ! ELEMENT COUNTER
1230         IF Y%+NY%(I%)<=512% GOTO 1280     ! BLOCK SIZE EXCEEDED
1240         UY%(I%)=UY%(I%)+512%-Y%           ! ACCUMULATE UNUSED BYTES
1250         Y%=0                              ! INITIALIZE BYTE COUNTER
1260         L%=L%+1%                          ! UPDATE BLOCK NO.
1270         GOTO 1230                         ! START FILLING THE NEW BLOCK
1280         Y%=Y%+NY%(I%)                     ! INCREMENT AN ELEMENT OF BYTES
1290       NEXT J%                             ! GET NEXT ELEMENT
1300 !
1310   EY%(I%)=Y%                ! ENDING BYTE NO.
1320   EL%(I%)=L%                ! ENDING BLOCK NO.
1330   UB%(K%)=UB%(K%)+UY%(I%)   ! ACCUMULATE TOTAL UNUSED BYTES
1340   TY%=TY%+SZ%(I%)           ! ACCUMULATE MINIMUM TOTAL BYTES REQUIRED
1350 !
1360 !                        ---- DISPLAY RESULTS ----
1370 !
1380   PRINT #1,USING Y$,TAB(5);MID(SQ$(K%),I%,1%);TAB(17);SY%(I%);
1390   PRINT #1,USING Y$,TAB(29);SL%(I%);TAB(44);EY%(I%);
1400   PRINT #1,USING Y$,TAB(56);EL%(I%);TAB(69);UY%(I%)
1410 !
1420 NEXT I%               ! GET NEXT ARRAY
1430 !
```

```
2000 !******************** DISPLAY TOTALS ************************
2010 !
2020 AL=EL%(N%)-1%+EY%(N%)/512
2030 TL=TY%/512
2040 AY%=AL*512
2050 PRINT #1, USING Y$,TAB(49);"TOTAL UNUSED BYTES";TAB(69);UB%(K%)
2060 PRINT #1
2070 PRINT #1, USING Y$,TAB(4);N%;" ARRAYS REQUIRING ";TY%;" BYTES IN ";
2080 PRINT #1, USING L$,TL;" BLOCKS"
2090 PRINT #1, USING Y$,TAB(14);"ACTUALLY USE ";AY%;" BYTES IN ";
2100 PRINT #1, USING L$,AL;" BLOCKS"
2110 PRINT #1
```

Program Listing is continued on next page

```
2120 TY%=0
2140 !
2150 !              ---- GET SUBSCRIPTS FOR NEXT SEQUENCE ----
2160 !
2170 NEXT C%
2180 NEXT B%
2190 NEXT A%
2200 !
3000 !**************** PICK BEST CHOICE ******************
3010 !
3020 MN%=UB%(1%) \ BC$=SQ$(1%)
3030 FOR K%=1% TO NF%
3040   IF UB%(K%)<MN% THEN MN%=UB%(K%) \ BC$=SQ$(K%) \ K1%=K%  ! PICK SMALLEST
3050 NEXT K%
3060 PRINT #1
3070 PRINT #1,TAB(25);"BEST CHOICE IS TO ORDER THE ARRAYS: ";BC$
3080 PRINT #1,TAB(25);"WHICH YIELDS";MN%;"UNUSED BYTES"
3090 PRINT #1
3100 FOR K%=K1%+1% TO NF%
3110   IF UB%(K%)=MN% THEN PRINT "OR ";SQ$(K%)
3120 NEXT K%
3130 CLOSE 1
3140 END
```

Special Consideration for OPEN Statements

It is often convenient to save the names of disk files as string elements in a virtual array. File names used in OPEN statements cannot contain the null characters which all virtual array strings use to fill up unassigned character positions, nor should they contain string functions such as LEFT. Before the contents of a virtual array string element can be used as a file name in an OPEN statement, you must assign its contents to a main memory string and delete the null characters. The main memory string (containing the desired file name without the null characters) should be used in the OPEN statement. Methods to delete null characters are found on page 2 of this lesson.

```
                    OUTPUT FROM "EXAM14"

          SUBSCRIPTS WHICH IDENTIFY ARRAYS
ARRAY NO.      1           2           3        (ORDER:  123)
 DIM #1, GX(1%,4%), E(10%,9%), Q$(10%)=64%   ! ARRAYS BEING TESTED

                          ORDER: 123
      ARRAY  STARTING     STARTING     ENDING      ENDING      UNUSED
      NO.      BYTE        BLOCK       BYTE        BLOCK       BYTES
 1              1           1           20          1           0
 2             21           1          392          2           4
 3            393           2          120          4          56
                                  TOTAL UNUSED BYTES          60

    3 ARRAYS REQUIRING  1604 BYTES IN    3.13 BLOCKS
        ACTUALLY USE    1664 BYTES IN    3.25 BLOCKS

                          ORDER: 132
      ARRAY  STARTING     STARTING     ENDING      ENDING      UNUSED
      NO.      BYTE        BLOCK       BYTE        BLOCK       BYTES
 1              1           1           20          1           0
 3             21           1          256          2          44
 2            257           2          112          4           0
                                  TOTAL UNUSED BYTES          44

    3 ARRAYS REQUIRING  1604 BYTES IN    3.13 BLOCKS
        ACTUALLY USE    1648 BYTES IN    3.22 BLOCKS

                          ORDER: 213
      ARRAY  STARTING     STARTING     ENDING      ENDING      UNUSED
      NO.      BYTE        BLOCK       BYTE        BLOCK       BYTES
 2              1           1          368          2           0
 1            369           2          388          2           0
 3            389           2          120          4          60
                                  TOTAL UNUSED BYTES          60

    3 ARRAYS REQUIRING  1604 BYTES IN    3.13 BLOCKS
        ACTUALLY USE    1664 BYTES IN    3.25 BLOCKS

                          ORDER: 312
      ARRAY  STARTING     STARTING     ENDING      ENDING      UNUSED
      NO.      BYTE        BLOCK       BYTE        BLOCK       BYTES
 3              1           1          192          2           0
 1            193           2          212          2           0
 2            213           2          72           4           4
                                  TOTAL UNUSED BYTES          4

    3 ARRAYS REQUIRING  1604 BYTES IN    3.13 BLOCKS
        ACTUALLY USE    1608 BYTES IN    3.14 BLOCKS

                          ORDER: 231
      ARRAY  STARTING     STARTING     ENDING      ENDING      UNUSED
      NO.      BYTE        BLOCK       BYTE        BLOCK       BYTES
 2              1           1          368          2           0
 3            369           2          64           4          16
 1             65           4          84           4           0
                                  TOTAL UNUSED BYTES          16

    3 ARRAYS REQUIRING  1604 BYTES IN    3.13 BLOCKS
        ACTUALLY USE    1620 BYTES IN    3.16 BLOCKS

                          ORDER: 321
      ARRAY  STARTING     STARTING     ENDING      ENDING      UNUSED
      NO.      BYTE        BLOCK       BYTE        BLOCK       BYTES
 3              1           1          192          2           0
 2            193           2          48           4           0
 1             49           4          68           4           0
                                  TOTAL UNUSED BYTES          0

    3 ARRAYS REQUIRING  1604 BYTES IN    3.13 BLOCKS
        ACTUALLY USE    1604 BYTES IN    3.13 BLOCKS
```


John Fluke Mfg. Co., Inc.
P.O. Box C9090, Everett, WA 98206
800-426-0361 (toll free) in most of U.S.A.
206-356-5400 from AK, HI, WA
206-356-5500 from other countries

Fluke (Holland) B.V.
P.O. Box 5053, 5004 EB, Tilburg, The Netherlands
Tel. (013) 673973, TELEX 52237
Phone or write for the name of your local Fluke representative.

Printed in U.S.A.

B0144A-10U8204/SE EN

FLUKE®

Technical Data

17XXA Software Information B0090

Stripping Remarks and Extra Spaces From Your Basic Program

Introduction

Many users have faced the dilemma of not having sufficient memory space for running large programs. One reason for this may be that the comments and spaces, required for a well-documented program, use a lot of memory. An additional problem caused by these comments and spaces is an increase in program run time.

One solution has been to run the program in modules. Although this solution works, it is not always the most desirable because it requires additional programming effort. Another solution has been manually to strip the extra spaces and comments from the program listing, thus allowing more program code to reside in main memory. Unfortunately, this solution also consumes and inordinate amount of programmer time and effort.

The purpose of this Software Bulletin is to present a program which will automatically strip remarks and unnecessary spaces from any BASIC program. The resulting program will occupy considerably less memory in most cases, and it will run faster because the comments and extra spaces do not have to be handled by the BASIC interpreter.

The following paragraphs will describe the equipment needed, the program and its limitations, how to enter and test the program, and the benefits of using the program.

Equipment Used

The only equipment needed to run the remark program is the minimum configuration 1720A Instrument Controller, and your program on Floppy or E-disk. Although the E-disk is not required to run the remark program, it will allow the program to run faster.

The Remark Program

The REMARK PROGRAM is written in Fluke 1720A BASIC. The program is divided into several sections. The first sections asks the operator for the input and output file names, then opens the files. The next section reads a line from the input file and strips all of the remarks from it. If a remark is the only thing that occupies a program line, that line will be deleted entirely. The next section of the program strips extra spaces from the remainder of the line and stores the line in the output file. It does not strip spaces from quoted fields, and it strips only invalid spaces from lines containing data statements.

Limitations

If your program contains statements that jump or refer to lines which contain only remarks, the stripped version of your program will produce errors when it encounters those statements. The reason for this is that the referenced lines will have been deleted by the remark program, producing an error when it is run. Therefore, it is a good idea to pre-check your program for all IF-THEN-ELSE, GOTO, GOSUB, STOP ON, ON GOTO, ON GOSUB, TRACE ON, and RESUME statements which refer to remark-only lines. Other program languages allow jumping to labels. Frequently, these labels are the remarks in the referenced line. Although Fluke BASIC allows jumps to these lines, the remark program does not compensate for the fact that such lines will have been deleted.

The remark program will not function properly if the input file which you identify is a lexical, binary, data, virtual array, system, (.SYS) or assembly language (.CIL) file. It will work properly only on BASIC programs which were originally saved on a mass storage device with the "SAVE" command using the BASIC editor. Other types of files will cause fatal or mysterious errors in the remark program.

The object program must have worked properly before running REMARK on it. Otherwise it may contain errors which REMARK assumes do not exist, causing REMARK to modify lines when it otherwise wouldn't. Thoroughly debug your program before running REMARK on it.

The portion of REMARK which deletes extra spaces may inadvertently create errors if the process creates or modifies BASIC keywords. For example, the statement FOR I=S TO P will be changed to FORI=STOP, creating the keyword STOP from the variable S, keyword TO, and variable P. Thus, it is a good idea to run your program through a thorough test cycle after stripping remarks and spaces from it. The Remark Stripping Program is now included in the BASIC startup disk, 1720A-902.

Using the Program

Running the Program

When you run the remark program, the display will ask whether you want to delete only remarks or both remarks and spaces, ask for the input file name (the name of the BASIC program you wish to strip remarks and extra spaces from), and ask the output file name (the name of the stripped version of your program). It will display each line as it is either deleted (as a remark only), or stripped of remarks and extra spaces. When the program is finished, it will display the words "JOB DONE!".

Strip Remarks Only

When the display prompts you to decide whether to strip only remarks or both remarks and spaces (see Figure 1) make one of the following entries:

1. R and press RETURN: strips remarks only;
2. Press RETURN only: strips both remarks and spaces.

```
Remark  Version  1.0
Strips remarks and extra spaces from BASIC programs; does not
correct jumps to remark-only lines deleted by this program !
R=delete remarks only; <RETURN> only= delete remarks and spaces
```

Figure 1. Initial display.

```
When you enter the input <filename.extension>,the .BAS extension
is not needed; invalid filenames or filetypes will cause errors.
Enter input filename =
```

Figure 2. Input file name request.

```
When you enter the output filename, CRT<RETURN> sends results to
the display only. .OUT adds to the input filename if you enter
<RETURN> only, or to output filename if you enter no extension.
Enter output file name =
```

Figure 3. Output file name request.

©1982, John Fluke Mfg. Co., Inc., all rights reserved

b. Test routine after only remarks are stripped.

```
1 DATA
2 DATA \ REM
3 DATA !
4 DATA REM
5 DATA \
100 DATA   45  ,  78  ,  'PRINT' , \REM , "REM" , !!!
102 DATA   DATA  ,  DATA  , 'DATA' , \
104 PRINT \ DATA 4 , " XY " , 7 ' X ' , 8 \ REM ! \ DATA
106 PRINT \ PRINT \ DATA   ,  2 , 23 ,    ,    ,   3
108 DATA   45  ,  78  ,  'PRINT' , \REM , "REM" , !!!
110 PRINT ' HELLO ' , " TODAY ", " 'I' " , ' A " ONE " 2 '
4000 REMOTE
4040 PRINT
4045 PRINT  'ONE',  ' "YES '
```

c. Test routine after both remarks and spaces are stripped.

```
1DATA
2DATA\ REM
3DATA!
4DATAREM
5DATA\
100DATA45  ,78  ,'PRINT',\REM ,"REM",!!!
102DATADATA  ,DATA  ,'DATA',\
104PRINT\DATA4 ," XY ",7 ' X ' ,8 \ REM ! \ DATA
106PRINT\PRINT\DATA,2 ,23 ,,,3
108DATA45  ,78  ,'PRINT',\REM ,"REM",!!!
110PRINT' HELLO '," TODAY "," 'I' ",' A " ONE " 2 '
4000REMOTE
4040PRINT
4045PRINT'ONE',' "YES '
```

Faster Run Time

A stripped version of your program will run faster than an unstripped version. For example, when the unstripped version of the remark program was run to strip the remarks and spaces from that same program, it required nearly 166 seconds of run time. However, when the stripped version of the remark program was run on the unstripped version, it required less than 140 seconds of run time. In both cases the floppy disk was the mass storage device. In this case, stripping the program before running it resulted in a 15% increase in speed.

PROGRAM	Mass Memory Blocks Used	Main Memory Used (Bytes)	Run Time *(Seconds) to strip REMARK.BAS of	
			Remarks	Remarks and Spaces
REMARK.BAS (ASCII, Unstripped)	31	15253	75.83	165.98
REMARK.BAL (lexical, Unstripped)	29	15253	75.83	165.98
REMREM.OUT (ASCII, Stripped of remarks)	14	6073	68.08	146.56
REMSPC.OUT (ASCII, Stripped of remarks and spaces)	8	2994	65.09	139.1
REMSPC.BAL (lexical, Stripped of remarks and spaces)	6	2994	65.09	139.1
*Using floppy as the system device.				

Table 2. STRIPPED and UNSTRIPPED REMARK program memory usage and run times.

For More Information

If you have questions regarding this Software Bulletin or any other Software applications for the 1720A Controller, contact your nearest Fluke sales representative or manufacturing facility.

FLUKE®

John Fluke Mfg. Co., Inc.
P.O. Box C9090, Everett, WA 98206
800-426-0361 (toll free) in most of U.S.A.
206-356-5400 from AK, HI, WA
206-356-5500 from other countries

Fluke (Holland) B.V.
P.O. Box 5053, 5004 EB, Tilburg, The Netherlands
Tel. (013) 673973, TELEX 52237
Phone or write for the name of your local Fluke representative.

Printed in U.S.A.

B0090C-10U8209/SE EN

FLUKE®

Technical Bulletin

1720A Software Information B0091

Converting Binary Data to Floating Point Numbers for 8500A, 8502A, & 8520A

Introduction

The 1720A Instrument Controller can accept readings from the 8500A series of Fluke Programmable Voltmeters in any format. However, only the ASCII (American Standard Code for Information Interchange) form can be used directly in computations with other real numbers. The various binary forms cannot.

The purpose of this Software Bulletin is to provide 1720A software routines which convert the binary formats to real numbers. The routines are provided with remarks to explain them. The actual statements which perform the conversion are minimal, however. One of the main objectives of the routines is to make the conversions as fast as possible. You can incorporate these routines as needed in your present software as explained in this bulletin.

Equipment Needed

The only equipment needed to implement the conversion routines is:

1720A Instrument Controller
Y1720A Programmer's Keyboard

To operate the programs in a system you also need:

8500A, 8502A, or 8520A Voltmeter
with IEEE-488 bus interface
Voltmeter test leads
IEEE-488 bus cable
The device under test

Background

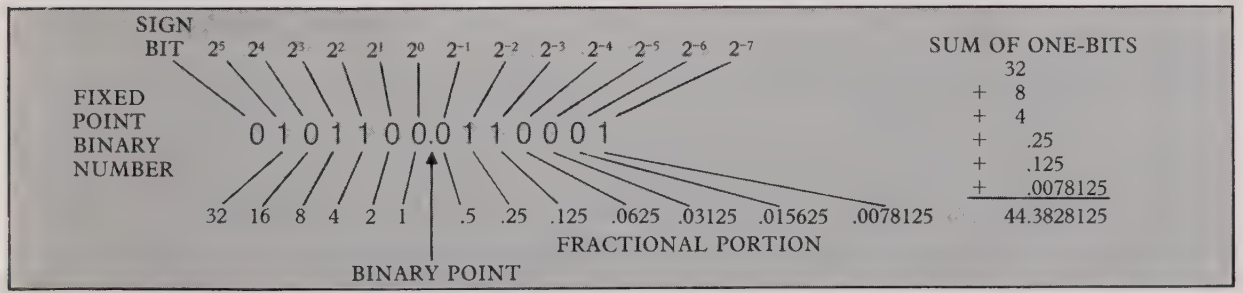
What are the Weights of Binary Digits in Binary Numbers with a Binary Point?

The value of digits to the left of the binary point (integer portion) equals the sum of individual one's multiplied by a positive power of 2. The value of the portion of the number to the right of the binary point (fractional portion) equals the sum of the individual one's multiplied by a negative power of 2. The power of 2 used depends on the bit position. This is shown in Example #1.

What does 2's Complement Mean?

2's complement is a method of representing signed binary numbers. In 2's complement fixed point form, the leftmost bit is a sign bit. If the number is in floating point format, a portion of it will have an exponent and the exponent will have its own sign bit. The value of the whole floating point number is the mantissa times ten-to-the-power-of-the-exponent for the 8500/8502, or two-to-the-power-of-the-exponent for the 8520A.

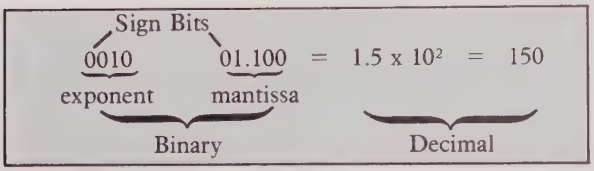
Example #1



All data, documentation, dialog, diagrams, suggestions, reports and/or other forms of media contained in this bulletin are intended to be informational in nature only. Implementation of such data to a user's application should ONLY be made after careful analysis by the user's own software experts. **John Fluke Mfg. Co., Inc., specifically disclaims all warranties on such information, express or implied, including but not limited to any warranty of merchantability, fitness, or adequacy for any particular purpose or use.**

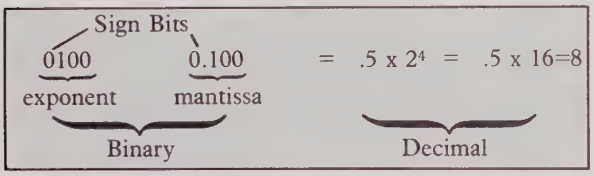
For example, in the below 8502-style binary number system, the exponent of the base 10 and the mantissa is 1.5:

Example #2



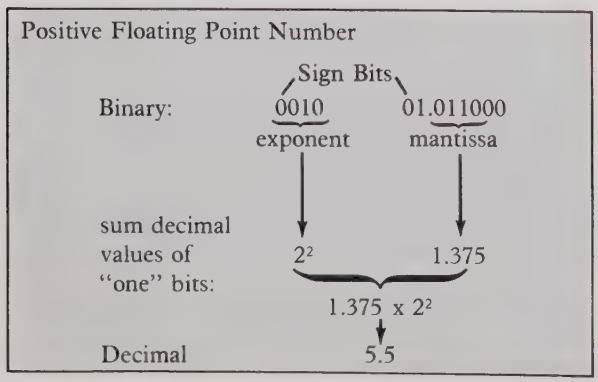
In the below example of the 8520-style binary number system, the exponent of the base 2 and the mantissa is .5:

Example #3

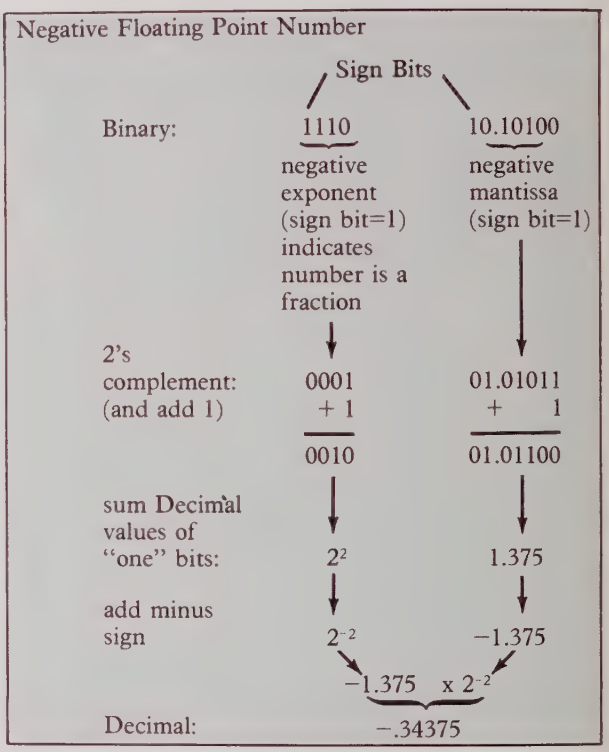


If the sign bit is a zero, the number is a positive value. If the sign bit is a one, the number is a negative value. The difference in the remaining bits is this: **you can determine the value of the base ten positive number by summing up the decimal values of the individual bits (see example #1); but to determine the value of the negative number you must complement the entire number (change all ones to zeros and vise versa) and add 1. Then sum the decimal values of the individual bits (as in example #1) and add a minus sign.** To convert the binary number back to negative form, complement all bits and add 1. Here are examples for both fixed point and floating point numbers.

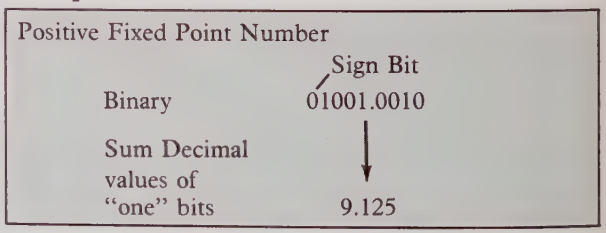
Example #4



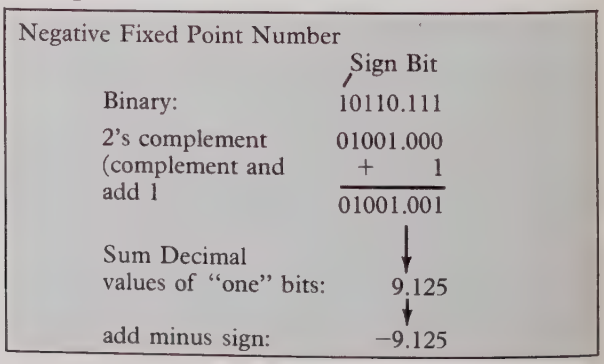
Example #5



Example #6



Example #7



1980, John Fluke Mfg. Co., Inc., all rights reserved

What does Sign-Magnitude mean?

Sign-magnitude is another way of representing binary numbers. In this form the leftmost bit is the sign bit. The remaining bits do not have to be complemented or modified in any way; they give the magnitude of the number directly. For example:

$1.011_{(2)} = -.375_{(10)}$, and $0.011_{(2)} = +.375_{(10)}$

A floating point number can have a 2's complement exponent and a sign-magnitude mantissa. Such is the case with 8520A 4-byte (normal mode) readings.

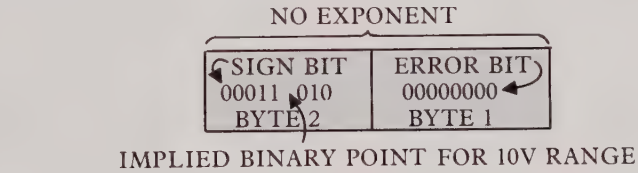
In converting input binary values, the conversion routines follow the philosophy of the above procedure. However, they deviate from the exact procedure in order to provide answers faster with complete accuracy. Further, the routines will only work on other controllers which do integer arithmetic.

General Description

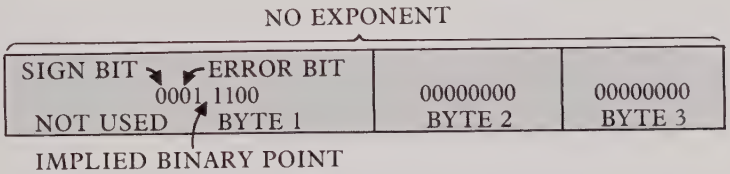
What are the Binary Formats?

The 8500A programmable voltmeters send readings over the IEEE-488 interface in the following binary formats. (Note: There are 8 bits per byte. All readings are in 2's complement form except the 8520A 4-byte binary which has sign-magnitude mantissa and 2's complement exponent.) The formats imply least significant bits to the right.

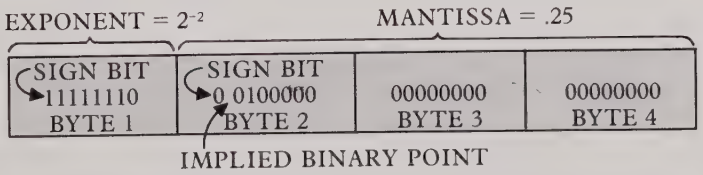
8520A High Speed Mode
2-Byte Binary
Fixed Point
(multiply by scale factor to get measured value) Reading=3.25



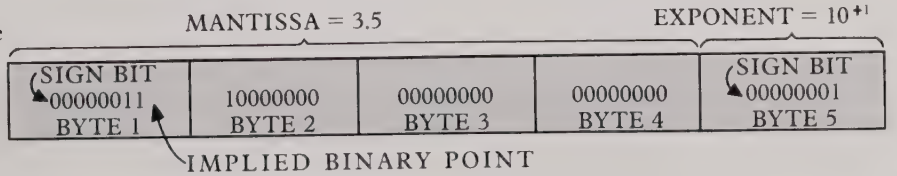
8502A High Speed Mode
3-Byte Binary
Fixed Point
(multiply by scale factor to get measured value) Reading=1.75



8520A Normal Mode
4-Byte Binary
Floating Point
(no scale factor)
(exponent is power of 2)
Reading=.0625



8500A/8502A Normal Mode
5-Byte Binary
Floating Point
(no scale factor)
(exponent is power of 10)
Reading=35



NOTE: See topic on "Difference Between Normal and High Speed Reading Modes" for correct Scale factors.

What does the Controller do with the Binary Values?

When the controller reads the binary values from the voltmeter, it cannot compute with them directly because the controller's internal format for numbers is different from the format received from the voltmeters. For example, the controller assigns 2 bytes of memory for each integer and 8 bytes for each floating point number (also called "real" number), while the voltmeters send 2, 3, 4, or 5 bytes. Since the binary formats of voltmeter readings are so varied and different, the controller must have a program or routine to convert the voltmeter readings into floating point numbers which it can recognize and use in computations.

The routines presented in this software bulletin perform those conversions, and multiply converted values by any necessary scale factors. They also minimize the time required to do the conversions. They then make the converted readings available to the user program in the form of real number (floating point) variables. The user program can use the values of these variables in subsequent computations.

What are the Scale Factors all about?

Readings taken from the voltmeters (8502A and 8520A) in high speed mode are in fixed point format. That is, there is no exponent as part of the reading, and the binary point is assumed always to be in the same position. Therefore, all readings taken in the high speed mode are in the same format, regardless of the range setting of the instrument. Because of this, each reading must be multiplied by a scale factor to adjust the reading for the range setting.

All readings from the 8502A in high speed mode are multiplied by 10 in the conversion routines to normalize them to the 10-Volt Range. Readings from the 8520A do not need to be multiplied by 10 first.

The conversion routines then multiply the 2-byte/3-byte readings by a scale factor which is based on the range setting of the respective voltmeter. The proper scale factors are given in the next topic. The user program must enter a scale factor parameter for the range being used into a local variable before calling the conversion routine.

What is the difference between Normal and High Speed Reading Modes?

In the normal modes, the 8500A, 8502A, and 8520A perform internal calculations to adjust the reading for the selected range. In the high speed mode, to save time, no internal adjustment is made. Therefore, the controller must make the adjustment. Normal mode readings include an exponent; high speed ones do not. Because of the greater number of bytes per reading, normal mode readings have more resolution than high speed ones.

8500A - The 8500A has no high-speed mode available on the IEEE-488 Bus. The reading sent is the actual measured value. It has a 5-byte binary format identical to the 8502A 5-byte format.

8502A - The 8502A has both modes. In normal mode the reading is the actual measured value in 5-byte binary format. In high speed mode the reading is in 3-byte binary format, and is not the actual measured value. To get the actual measured value, the controller must multiply the reading by 10.

The 8502A can take high speed readings on all functions: DC Volts (VDC), AC Volts (VAC), DC Current (IDC), AC Current (IAC), and Ohms. When taking ohms measurements, scale factors must be uniquely computed, then multiplied by the converted measured value. Although the somewhat complex method for computing ohms scale factors is not discussed in this bulletin, complete information is provided by Application Bulletin 25, Appendix A.

Regardless of the type of measurement being taken, (when in binary mode) the reading must be multiplied by a scale factor to yield the actual measured value. The scale factor is computed by the formula SF=10 x RF, where RF (Range Factor) for the different functions is given below. The 3-byte conversion routine automatically performs the times-10 multiplication of the Range Factor once you have put the range factor into local variable RF.

Functions/Range			
VDC	**VAC**	**IAC**	**Scale Factor**
100 mV	---	100 uA	1/64
1V	1V	1 mA	1/8
10V	10V	10 mA	1
100V	100V	100 mA	8
1 kV	1 kV	1A	64

Functions/Range	
IDC	**Scale Factor**
100 uA	−1/64
1 mA	−1/8
10 mA	−1
100 mA	−8
1A	−64

NOTE: Ohms scale factor must be computed from prior ohms measurements. Refer to AB25 for details.

In the high-speed mode, the 7th bit of byte 1 is not used. The 6th bit indicates an error condition (overrange) if different from the sign (8th) bit (absolute value of reading is greater than 2.0).

8520A - The 8520A has both modes. In normal mode the reading is the actual measured value in 4-byte binary format. The reading has a 2's complement exponent and sign-magnitude mantissa. Furthermore, if the exponent equals 128, the entire value equals zero. In high speed mode the reading is in 2-byte binary format, and is the actual measured value only if the 8520A is set to the 10-volt range. On any other range, the reading must be multiplied by below scale factors for DC volts (done by the conversion routine). In this mode, the rightmost bit of byte 1 will equal a 1 if the measured value is beyond the selected range. Scale-factors for ohms measurements are different from those for voltage measurements. Negative scale factors are used to give positive results.

Resistance Range	Scale Factor
10 ohms	-1.5625
100 Ohms	-12.5
1K Ohm	-125
10K Ohms	-1250

DC Volts Range	Scale Factor
100 mV	1/64
1V	1/8
10V	1
100V	8
1000V	64

The Conversion Routines

What are They?

Tables 1 through 4 present listings of subroutines written in Fluke BASIC for the 1720A Instrument Controller. The routines convert voltmeter binary readings which have been input as integer arrays to floating point values recognizable by the 1720A, as follows:

Table 1. 8520A High Speed Mode 2-byte Binary

```
10000 !**********************   Subroutine  2BYTE   ***************************
10002 !
10004 !   Version 1.0      20 May 1980   Fluke Control Products Marketing
10006 !
10008 !                     BASIC V1.0, FDOS 1.1
10010 !
10012 !   DESCRIPTION:  2BYTE will convert an integer array < IA% > read
10014 ! in from the 8520A in high speed binary mode (2-byte transfers)
10016 ! into 1720A format and put the answers in result array < RA >.
10018 !   If the error bit is set by the 8520A then that reading is set
10020 ! to zero.
10022 !
10024 !   Format of the 2-byte reading is:
10026 !
10028 ! 1st byte:  bit 0    = +/- overrange error bit
10030 !            bits 1-7 = lower ordered fraction bits (2^-10 TO 2^-4)
10032 ! 2nd byte:  bits 0-2 = higher ordered fraction bits (2^-3 TO 2^-1)
10034 !            Implied Binary Point
10036 !            bits 3-6 = integer bits
10038 !            bit 7    = sign bit
10040 !            bit 8    = EOI bit (end of reading; must be masked out)
10042 !
10044 !   Variables to be initialized before calling:
10046 !
10048 !       IA% = input array with concatenated bytes of all readings
10050 !       IM% = size of input array
10052 !       RA  = result array
10054 !       RF  = range scale factor as follows:
10056 !
10058 !             1=10-V ; 1/64=100-mV ; 1/8=1-V ; 8=100-V ; 64=1-kV
10060 !          -1.5625=10-Ohm ; -12.5=100-Ohm ; -125=1-kOhm ; -1250=10-kOhm
10062 !
```

Table 1. 8520A High Speed Mode 2-byte Binary (Con't.)

```
10064 !   Local Variables:
10066 !
10068 !       B1%,B2%       = bytes 1 and 2 of the readings
10070 !       C1%,C2%       = integer contants for complementing
10072 !       C1,C2         = real constants for the binary conversion
10074 !       I%            = loop counter
10076 !       IS%           = sign of number
10078 !
10100 C1% = 255%                                      ! constant for 1's complement
10110 C2% = 256%                                      ! constant for 2's complement
10120 C1  = 8/RF                                      ! constant for byte 1
10130 C2  = 2048/RF                                   ! constant for byte 2
10140 !
10150 FOR I% = 0% TO IM% STEP 2%                      ! conversion loop
10160 B1% = IA%(I%)                                   ! setup byte 1
10170 IF B1% AND 1% THEN RA(I%/2%)=0 \GOTO 10240      ! error has occurred
10180 B2% = IA%(I%+1%) AND C1%                        ! setup byte 2, strip EOI
10190 IF B2%<128% THEN IS% =1% \ GOTO 10230           ! check for negative readings
10200 IS% = -1%                                       ! set sign to negative
10210 B1% = C2%-B1%                                   ! complement byte 1
10220 B2% = C1%-B2%                                   ! complement byte 2
10230 RA(I%/2%) = (B2%/C1 + B1%/C2) * IS%             ! compute readings
10240 NEXT I%                                         ! loop
10250 !
10260 RETURN
```

Table 2. 8502A High Speed Mode 3-byte Binary

```
11000 !******************   Subroutine 3BYTE   ******************************
11002 !
11004 !   Version 1.0       20 May 1980     Fluke Control Products Marketing
11006 !
11008 !           System Software: BASIC V1.0, FDOS V1.1
11010 !
11012 !   DESCRIPTION:   3BYTE will convert an integer array <IA%> read in
11014 ! from the 8502A DVM in high speed mode (3-byte binary transfer) to
11016 ! 1720A format, and store the answers in result array <RA>.
11018 !   If the error bit is set by the 8502A then that reading is
11020 ! set to zero.
11022 !
11024 !   Format of the 3-byte readings is:
11026 !
11028 ! 1st byte:  bits 0-3 = upper order fraction bits (2^-4 to 2^-1)
11030 !            Implied Binary Point
11032 !            bit 4    = integer bit
11034 !            bit 5    = error bit (set if absolute reading -> 2.0)
11036 !            bit 6    = not used
11038 !            bit 7    = sign bit
11040 ! 2nd byte:  bits 0-7 = middle order fraction bits (2^-12 to 2^-5)
11042 ! 3rd byte:  bits 0-7 = lower order fraction bits (2^-20 TO 2^-13)
11044 !            bit 8    = EOI bit (end of readings; must be masked out)
```

Table 2. 8502A High Speed Mode 3-byte Binary (Con't.)

```
11046 !
11048 !   Variables to be initialized before calling:
11050 !
11052 !      IA% = input array with concatenated bytes of readings
11054 !      IM% = size of input array
11056 !      RA  = result array
11058 !      RF  = range scale factor as follows:
11060 !            1=10-V ; 1/64=100-mV ; 1/8=1-V ; 8=100-V ; 64=1-kV
11062 !
11064 !    Local variables:
11066 !
11068 !      B1%,B2%,B3% = bytes 1, 2, and 3
11070 !      C1%,C2%     = integer constants for complementing
11072 !      C1,C2,C3    = real constants for converting
11074 !      I%          = loop counter
11076 !      IS%         = sign of number
11078 !
11100 C1% = 255%                                    ! constant for 1's complement
11110 C2% = 256%                                    ! constant for 2's complement
11120 C1  = 16/(10*RF)                              ! constant for byte 1
11130 C2  = 4096/(10*RF)                            ! constant for byte 2 (2^12)
11140 C3  = 1048576/(10*RF)                         ! constant for byte 3 (2^20)
11150 !
11160 FOR I% = 0% TO IM% STEP 3%                    ! conversion loop
11170 B1% = IA%(I%)                                 ! setup byte 1
11180 B2% = IA%(I%+1%)                              ! setup byte 2
11190 B3% = IA%(I%+2%) AND C1%                      ! setup byte 3; strip EOI
11200 IF B1%<128% THEN IS% =1% \ GOTO 11250         ! check for negative readings
11210 IS% = -1%                                     ! set sign to negative
11220 B1% = C1%-B1%                                 ! complement byte 1
11230 B2% = C1%-B2%                                 ! complement byte 2
11240 B3% = C2%-B3%                                 ! complement byte 3
11250 IF B1% AND 32% THEN RA(I%/3%)=0 \GOTO 11270   ! error has occurred
11260 RA(I%/3%)= (B1%/C1 + B2%/C2 + B3%/C3) *IS%    ! compute result
11270 NEXT I%                                       ! loop
11280 !
11290 RETURN
```

Table 3. 8520A 4-byte Binary

```
12000 !*****************   Subroutine 4BYTE   ******************************
12002 !
12004 !   Version 1.0       20 May 1980     Fluke control Products Marketing
12006 !
12008 !         System Software:   BASIC V1.0, FDOS V1.1
12010 !
12012 !   DESCRIPTION:  4BYTE will convert an integer array <IA%> read in
12014 ! from the 8520A in 4-byte mode to 1720A format and store the answers
12016 ! in result array <RA>.
12018 !
12020 !   Format of the 4-byte readings is:
12022 !
12024 ! 1st byte:  bits 0-6 = exponent (2^X) bits 2^0 to 2^6
12026 !            bit 7    = exponent sign bit
```

Table 3. 8520A 4-byte Binary (Con't.)

```
12028 ! 2nd byte:  bits 0-6 = upper order fraction bits (2^-7 to 2^-1)
12030 !            Mantissa Implied Binary Point
12032 !            bit 7    = mantissa sign bit
12034 ! 3rd byte:  bits 0-7 = middle order fraction bits (2^-15 to 2^-8)
12036 ! 4th byte:  bits 0-7 = lower order fraction bits (2^-23 to 2^-16)
12038 !            bit 8    = EOI bit (end of reading; must be masked out)
12040 !
12042 !   Variables to be initialized before calling:
12044 !
12046 !      IA% = input array with concatenated bytes of all readings
12048 !      IM% = size of input array
12050 !      RA  = result array
12052 !
12054 !   Local variables:
12056 !
12058 !      B1%,B2%,B4% = bytes 1,2,4 (byte 1 is exponent)
12060 !      C1%,C2%     = constants for complementing/masking
12062 !      C1,C2,C3    = constants for the binary conversion
12064 !      EX          = computed exponent
12066 !      I%          = loop counter
12068 !      IS%         = sign of number
12070 !
12100 C1% = 255%                                    ! 8-bit mask (1's complement)
12110 C2% = 256%                                    ! constant for 2's complement
12120 C2  = 128                                     ! constant for byte 2 (2^7)
12130 C3  = 32768                                   ! constant for byte 3 (2^15)
12140 C4  = 8388608                                 ! constant for byte 4 (2^23)
12150 !
12160 FOR I% = 0% TO IM% STEP 4%                    ! conversion loop
12170 B1% = IA%(I%)                                 ! setup byte 1
12180 B2% = IA%(I%+1%)                              ! setup byte 2
12190 B4% = IA%(I%+3%) AND C1%                      ! setup byte 4; strip EOI
12200 IF B2%<128% THEN IS% = 1% \ GOTO 12230        ! check for negative readings
12210 IS% = -1%                                     ! set sign to negative
12220 B2% = B2% AND 127%                            ! mask sign bit
12230 IF B1%>127% THEN B1% = (C2%-B1%) *-1%         ! complement exponent
12240 IF B1% THEN EX = 2%^B1%*IS%  ELSE EX=0%       ! compute exponent
12250 RA(I%/4%) = (B2%/C2 +IA%(I%+2%)/C3 +B4%/C4) *EX      ! compute result
12260 NEXT I%                                       ! loop
12270 !
12280 RETURN
```

Table 4. 8500A or 8502A 5-byte Binary

```
13000 !*******************   Subroutine 5BYTE   ********************************
13002 !
13004 !   Version 1.0      20 May 1980    Fluke Control Products Marketing
13006 !
13008 !            System Software:   BASIC V1.0, FDOS V1.1
13010 !
13012 !   DESCRIPTION:  5BYTE will convert an integer array <IA%> read in
13014 ! from the 8500A or 8502A DVM in 5-byte mode to 1720A format, and
```

Table 4. 8500A or 8502A 5-byte Binary (Con't.)

```
13016 ! store the answers in result array <RA>.
13018 !
13020 !   Format for the 5-byte readings is:
13022 !
13024 ! 1st byte:  bits 0-6 = mantissa integer bits 2^0 thru 2^6
13026 !             bit 7   = mantissa sign bit
13028 !             Implied Binary Point
13030 ! 2nd byte:  bits 0-7 = mantissa high order fraction bits (2^-8 to 2^-1)
13032 ! 3rd byte:  bits 0-7 = mantissa medium fraction bits (2^-16 to 2^-9)
13034 ! 4th byte:  bits 0-7 = mantissa lower  fraction bits (2^-24 to 2^-17)
13036 ! 5th byte:  bits 0-7 = exponent (10^X) bits 2^-0 to 2^-6)
13038 !             bit 7   = exponent sign bit
13040 !             bit 8   = EOI bit (end of readings; must be masked out)
13042 !
13044 !   Variables to be initialized before calling:
13046 !
13048 !       IA% = input array with concatenated bytes of all readings
13050 !       IM% = size of input array
13052 !       RA  = result array
13054 !
13056 !   Local variables are:
13058 !
13060 !       B1% - B5%   = bytes 1 through 5 (byte 5 is exponent)
13062 !       C1%,C2%     = constants for complementing & conversion
13064 !       C3,C4       = constants for the binary conversion
13066 !       EX          = computed exponent
13068 !       I%          = loop counter
13070 !       IS%         = sign of number
13072 !
13100 C1% = 255%                                        ! constant for 1's complement
13110 C2% = 256%                                        ! constant for 2's complement
13120 C2  = 256                                         ! constant for byte 2 (2^8)
13130 C3  = 65536                                       ! constant for byte 3 (2^16)
13140 C4  = 16777216                                    ! constant for byte 4 (2^24)
13150 !
13160 FOR I% = 0% TO IM% STEP 5%                        ! conversion loop
13170 B1% = IA%(I%)       \   B2% = IA%(I%+1%)          ! setup bytes 1 and 2
13180 B3% = IA%(I%+2%)    \   B4% = IA%(I%+3%)          ! setup bytes 3 and 4
13190 B5% = IA%(I%+4%) AND C1%                          ! setup byte 5; strip EOI
13200 IF B1%<128% THEN IS% =1% \ GOTO 13240             ! check for negative readings
13210 IS% = -1%                                         ! set sign to negative
13220 B1% = C1%-B1%       \   B2% = C1%-B2%             ! complement bytes 1 and 2
13230 B3% = C1%-B3%       \   B4% = C2%-B4%             ! complement bytes 3 and 4
13240 IF B5%>127% THEN B5% = (C2%-B5%) * -1%            ! complement exponent
13250 EX = 10%^B5% * IS%                                ! compute exponent
13260 RA(I%/5%) = (B1%+B2%/C2+B3%/C3+B4%/C4)*EX         ! compute result
13270 NEXT I%                                           ! loop
13280 !
13290 RETURN
```

How do they work with My Program?

Figure 1 shows the necessary structure of your program using the 8520A 2-byte (High Speed Mode) conversion routine. Refer to the figure for the following discussion.

The conversion routines function as callable subroutines. That is, your program accesses one of them with the GOSUB statement. Each routine assumes that your program has read one or more readings of the appropriate number of bytes each from the voltmeter, and concatenated them into the elements of an input array (IA%). Your program must also have dimensioned the input array and the result array for the number of readings taken and selected the correct scale factor.

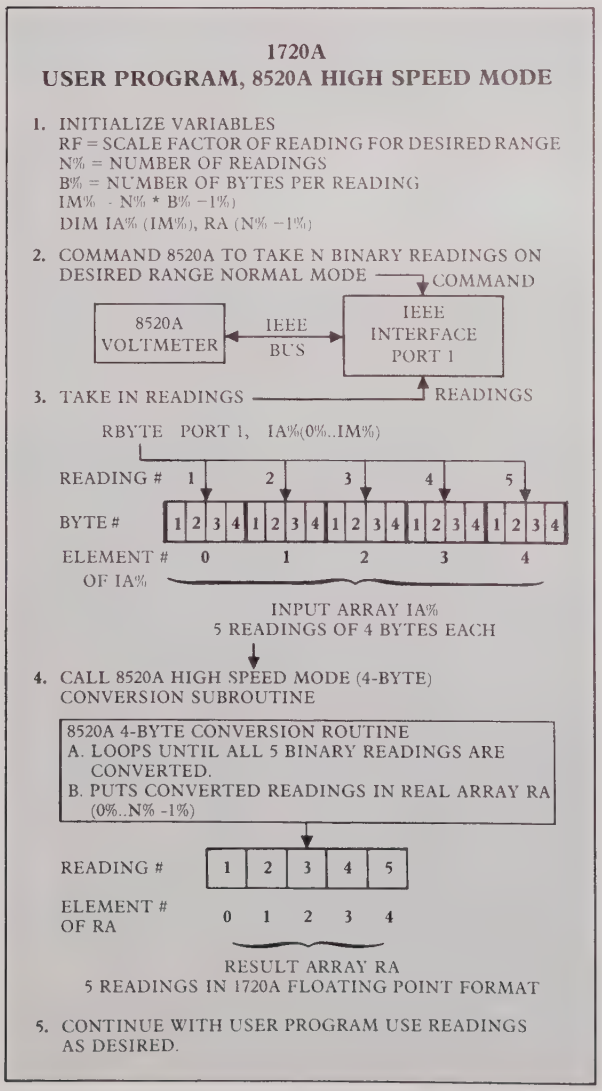

1720A
USER PROGRAM, 8520A HIGH SPEED MODE
1. INITIALIZE VARIABLES
RF = SCALE FACTOR OF READING FOR DESIRED RANGE
N% = NUMBER OF READINGS
B% = NUMBER OF BYTES PER READING
IM% = N% * B% -1%)
DIM IA% (IM%), RA (N% -1%)
2. COMMAND 8520A TO TAKE N BINARY READINGS ON DESIRED RANGE NORMAL MODE
COMMAND
8520A VOLTMETER
IEEE BUS
IEEE INTERFACE PORT 1
3. TAKE IN READINGS
READINGS
RBYTE PORT 1, IA%(0%..IM%)
READING # 1 2 3 4 5
BYTE # 1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4
ELEMENT # OF IA% 0 1 2 3 4
INPUT ARRAY IA%
5 READINGS OF 4 BYTES EACH
4. CALL 8520A HIGH SPEED MODE (4-BYTE) CONVERSION SUBROUTINE
8520A 4-BYTE CONVERSION ROUTINE
A. LOOPS UNTIL ALL 5 BINARY READINGS ARE CONVERTED.
B. PUTS CONVERTED READINGS IN REAL ARRAY RA (0%..N% -1%)
READING # 1 2 3 4 5
ELEMENT # OF RA 0 1 2 3 4
RESULT ARRAY RA
5 READINGS IN 1720A FLOATING POINT FORMAT
5. CONTINUE WITH USER PROGRAM USE READINGS AS DESIRED.

Figure 1. Program Structure to Use Conversion Routines

1. Set variable IM%=N%*B%-1%. This sets up IM% to be the size of integer array IA%; N%= the number of readings to be taken; B%= the number of bytes per reading; 1 is subtracted because array element numbers start at zero.
2. Set variable RF to the scale factor for the range selected This is needed only in 8520A 2-byte and 8502A 3-byte modes.
3. Dimension the readings' input array IA% for the total number of bytes to be read, and the conversion routine result array RA for the number of readings to be taken. The statement DIM IA% (IM%), RA (N%-1%) does this.
4. Program the voltmeter to range and function. Then, read the desired number (N%) of readings into input array IA% (0%..IM%). Readings are concatenated in the array.
5. Call the conversion routine. GOSUB XXXXX does this where XXXXX is the line number of the first executable statement of the subroutine.

The conversion routine takes the bytes from the elements of array IA%, converts them to real number (floating point) format, and puts results in array RA.

If you are using either the 8502A or the 8520A in high speed mode, the conversion routine then multiplies the elements of array RA by the appropriate scale factor to get the correct measurement results. The scale factors are given in the general description. Finally, the conversion routine returns to where your program left off.

The number of readings the conversion routine will handle is determined by the dimensions of the input and result arrays, and thus by available memory. If insufficient memory is available for the number of readings you want to take, your program can set up the result array (RA) as a virtual array. As such, the elements of array RA will be stored in mass memory (Disk or E-Disk). The integer array IA% (input array) cannot be a virtual array because of the way the RBYTE statement is implemented inside the 1720A. The procedure for using virtual arrays is described in the 1720A Programmer's Manual.

How do I enter the Conversion Routine?

1. Select the conversion routine for your application.
2. Ensure that the variables used by the routine selected are not the same as variables intended for other purposes in your program. If they are, change the ones which are easiest to change so that no conflicts occur. If you intend to use more than one conversion routine in your overall program, you may need to rename the variables IA%, IM%, RA and RF. You do not need to change what the listings refer to as LOCAL VARIABLES (unless they interfere with your program) because they are initialized every time the routine is entered.
3. Wherever you want your program to call the subroutine, key in a GOSUB nnnnn statement, where the nnnnn is the starting line number (first executable line which is not a remark) of the subroutine.

Example Programs

Tables 5 through 8 list example programs which use the Binary Conversion Routines in Tables 1 through 4. The programs set up the 8502/8520 DVMs to take 100 voltage readings in the High Speed or Normal mode, put the bytes read into an input array, and display the results. The programs assume that the conversion subroutines are merged into the programs as described in the listing heading. Use the merge command (/M) in the File Utility Program to accomplish this (e.g., DVM2=8520 2, 2 BYTE/M).

Note that the example DVM commands shown between the asterisk lines are examples only. They may not be suitable for your application.

Errors which may occur are overflow error 0 if main memory has too little room for the array of input bytes, and error 306, meaning your mass storage medium has too little room for the result array. To correct the first error reload the program (it should not be merged with any other programs or subroutines than those mentioned in the listing heading). To correct the second error, insert a disk with more space, or purge some of the files.

Other errors may result from incorrect DVM command statements. Consult the DVM manual for correct procedures.

Table 5. 2-byte User Program

```
 1 !                          Program 8520 2
 2 !
 3 !        Version 1.0       20 May 1980      Fluke Control Products Marketing
 4 !
 5 !            System Software:   FDOS V 1.1, BASIC V 1.0
 6 !
10 ! DESCRIPTION:
12 !
14 !     This program will take 100 readings from the 8520A DVM in the 2-byte
16 ! mode (100-volt range), convert them to 1720A format using the 2BYTE sub-
18 ! routine, and display them.  The statements used to command the DVM are
20 ! EXAMPLES only, and are likely not to work in your application.  Key in
22 ! the appropriate statements in their place before running the program.
24 !     The 2BYTE routine is assumed to be stored at line 10000.
26 !
28 !
1000 !******************   Main Program   ************************************
1001 !
1010 N% = 100%                              ! number of readings to take
1020 B% = 2%                                ! number of bytes per reading
1030 RF = 8                                 ! range factor for 100-V range
1040 IM% = (N% * B%) -1%                    ! size of input array
1050 DIM IA%(IM%), RA(N%-1%)                ! dimension arrays
1060 !
1070 VM% = 2%                               ! DVM device number
1080 P% = 0%                                ! IEEE 488 port number
1090 INIT PORT P%                           ! send IFC and REN to bus
```

Table 5. 2-byte User Program (Con't.)

```
1100 CLEAR PORT P%                                  ! clear bus devices
1110 !
1120 !****************** EXAMPLE STATEMENTS*************************************
1130 ! PRINT @VM%,'VR3ISTO?'                        ! command DVM to take readings   *
1140 ! INPUT @VM%, OB$                              ! clear DVM output buffer        *
1150 ! RBYTE PORT P%, IA%(0%..IM%)                  ! input readings                 *
1160 !                                                                               *
1170 !     NOTE:  The above is an example and is given for demonstration             *
1180 ! purposes only.  Because your application may differ, careful                  *
1190 ! considerations is advised before any implementation.  Refer to the           *
1200 ! DVM manual for exact details on how to command the DVM to perform            *
1210 ! its functions.                                                               *
1220 !*******************************************************************************
1230 !
1240 GOSUB 10100                                    ! call 2BYTE conversion routine
1250 !
1260 FOR I%=1% TO N%                                ! display loop
1270 PRINT I%, RA(I%-1%)                            ! display readings
1280 NEXT I%                                        ! loop till finished
1290 !
1300 END
1310 !
1320 !
10000 !                     Subroutine  2BYTE
```

Table 6. 3-byte User Program

```
1 !                          Program 8502 3
2 !
3 !        Version 1.0       20 May 1980      Fluke Control Products Marketing
4 !
5 !         System Software:   FDOS V 1.1, BASIC V 1.0
6 !
10 ! DESCRIPTION:
12 !
14 !     This program will take 100 readings from the 8502A DVM in the 3-byte
16 ! mode (1-KVolt range), convert them to 1720A format using the 3BYTE sub-
18 ! routine, and display them.  The statements used to command the DVM are
20 ! EXAMPLES only, and are likely not to work in your application.  Key in
22 ! the appropriate statements in their place before running the program.
24 ! Note that the DVM must be externally triggered in the 3-byte mode.
26 !     The 3BYTE routine is assumed to be stored at line 11000.
28 !
30 !
1000 !******************   Main Program   ***********************************
1001 !
1010 N% = 100%                                      ! number of readings to take
1020 B% = 3%                                        ! number of bytes per reading
1030 RF = 64                                        ! range factor for 1KV range
1040 IM% = (N% * B%) -1%                            ! size of input array
1050 DIM BC%(2%), IA%(IM%), RA(N%-1%)               ! dimension arrays
1060 !
```

Table 6. 3-byte User Program (Con't.)

```
1070 P% = 0%                                    ! IEEE 488 port number
1080 VM% = 2%                                   ! DVM device number
1090 BC%(0%) = VM% + 544%                       ! DVM listen address
1100 BC%(1%) = ASCII('?')                       ! DVM trigger command
1110 BC%(2%) = VM% + 576%                       ! DVM talk address
1120 INIT PORT P%                               ! send IFC and REN to bus
1130 CLEAR PORT P%                              ! clear bus devices
1140 WAIT 3500                                  ! wait for DVM to clear
1150 !
1160 !****************** EXAMPLE STATEMENTS*********************************
1170 ! PRINT @VM%,'VR4BT2,!'                    ! command DVM to take readings  *
1180 ! INPUT @VM%, OB$                          ! clear DVM output buffer       *
1190 ! RBYTE PORT P%,(WBYTE PORT P%,BC%(0%..2%)) IA%(0%..IM%);3 ! input rdgs *
1200 !                                                                         *
1210 !     NOTE:  The above is an example and is given for demonstration       *
1220 ! purposes only.  Because your application may differ, careful           *
1230 ! considerations is advised before any implementation.  Refer to the     *
1240 ! DVM manual for exact details on how to command the DVM to perform      *
1250 ! its functions.                                                         *
1260 !*************************************************************************
1270 !
1280 GOSUB 11100                                ! call 3BYTE conversion routine
1290 !
1300 FOR I%=1% TO N%                            ! display loop
1310 PRINT I%, RA(I%-1%)                        ! display readings
1320 NEXT I%                                    ! loop till finished
1330 !
1340 END
1350 !
1360 !
11000 !******************   Subroutine 3BYTE   *******************************
```

Table 7. 4-byte User Program

```
1 !                          Program 8520 4
2 !
3 !        Version 1.0       20 May 1980      Fluke Control Products Marketing
4 !
5 !              System Software:   FDOS V 1.1, BASIC V 1.0
6 !
10 ! DESCRIPTION:
12 !
14 !     This program will take 100 readings from the 8520A DVM in the 4-byte
16 ! mode, convert them to 1720A format using the 4BYTE subroutine, and
18 ! display them.  The statements used to command the DVM are examples only
20 ! and are likely not to work in your application.  Key in the appropriate
22 ! statements in their place before running the program.
24 !     The 4BYTE routine is assumed to be stored at line 12000.
26 !
28 !
```

Table 7. 4-byte User Program (Con't.)

```
1000 !********************   Main Program   ***********************************
1001 !
1010 N% = 100%                                  ! number of readings to take
1020 B% = 4%                                    ! number of bytes per reading
1030 IM% = (N% * B%) -1%                        ! size of input array
1040 VM% = 2%                                   ! DVM device number
1050 P% = 0%                                    ! IEEE 488 port number
1060 DIM IA%(IM%), RA(N%-1%)                    ! dimension arrays
1070 !
1080 INIT PORT P%                               ! send IFC and REN to bus
1090 CLEAR PORT P%                              ! clear bus devices
1100 !
1110 !******************* EXAMPLE STATEMENTS*************************************
1120 ! PRINT @VM%,'DOI4TO?'                     ! command DVM to take readings   *
1130 ! INPUT @VM%, OB$                          ! clear DVM output buffer        *
1140 ! RBYTE PORT P%, IA%(0%..IM%)              ! input readings                 *
1150 !                                                                           *
1160 !     NOTE:  The above is an example and is given for demonstration     *
1170 ! purposes only.  Because your application may differ, careful           *
1180 ! considerations is advised before any implementation.  Refer to the     *
1190 ! DVM manual for exact details on how to command the DVM to perform      *
1200 ! its functions.                                                         *
1210 !***************************************************************************
1220 !
1230 GOSUB 12100                                ! call 4BYTE conversion routine
1240 !
1250 FOR I%=1% TO N%                            ! display loop
1260 PRINT I%, RA(I%-1%)                        ! display readings
1270 NEXT I%                                    ! loop till finished
1280 !
1290 END
1300 !
1310 !
12000 !*****************   Subroutine 4BYTE   ********************************
```

Table 8. 5-byte User Program

```
1 !                        Program 8502 5
2 !
3 !        Version 1.0       20 May 1980      Fluke Control Products Marketing
4 !
5 !            System Software:   FDOS V 1.1, BASIC V 1.0
6 !
10 ! DESCRIPTION:
12 !
14 !      This program will take 100 readings from the 8502A DVM in the 5-byte
16 ! mode (1-KVolt range), convert them to 1720A format using the 5BYTE sub-
18 ! routine, and display them.  The statements used to command the DVM are
20 ! EXAMPLES only, and are likely not to work in your application.  Key in
22 ! the appropriate statements in their place before running the program.
24 !      The 5BYTE routine is assumed to be stored at line 13000.
```

Table 8. 5-byte User Program (Con't.)

```
26 !
28 !
1000 !*********************    Main Program    ***********************************
1001 !
1010 N% = 100%                                  ! number of readings to take
1020 B% = 5%                                    ! number of bytes per reading
1030 IM% = (N% * B%) -1%                        ! size of input array
1040 DIM IA%(IM%), RA(N%-1%), T%(0%)            ! dimension arrays
1050 !
1060 P% = 0%                                    ! IEEE 488 port number
1070 VM% = 2%                                   ! DVM device number
1080 T%(0%) = VM% + 576%                        ! DVM talk address
1090 INIT PORT P%                               ! send IFC and REN to bus
1100 CLEAR PORT P%                              ! clear bus devices
1110 WAIT 3500                                  ! wait for DVM to clear
1120 !
1130 !******************* EXAMPLE STATEMENTS*************************************
1140 ! PRINT @VM%,'VR4BSOT?'                    ! command DVM to take readings   *
1150 ! WBYTE PORT P%, T%(0%)                    ! command DVM to talk            *
1160 ! RBYTE PORT P%, IA%(0%..IM%)              ! input rdgs                     *
1170 !                                                                           *
1180 !     NOTE:  The above is an example and is given for demonstration         *
1190 ! purposes only.  Because your application may differ, careful             *
1200 ! considerations is advised before any implementation.  Refer to the       *
1210 ! DVM manual for exact details on how to command the DVM to perform        *
1220 ! its functions.                                                           *
1230 !***************************************************************************
1240 !
1250 GOSUB 13100                                ! call 5BYTE conversion routine
1260 !
1270 FOR I%=1% TO N%                            ! display loop
1280 PRINT I%, RA(I%-1%)                        ! display readings
1290 NEXT I%                                    ! loop till finished
1300 !
1310 END
1320 !
1330 !
13000 !******************   Subroutine 5BYTE   ********************************
```

John Fluke Mfg. Co., Inc.
P.O. Box C9090, Everett, WA 98206
800-426-0361 (toll free) in most of U.S.A.
206-356-5400 from AK, HI, WA
206-356-5500 from other countries

Fluke (Holland) B.V.
P.O. Box 5053, 5004 EB, Tilburg, The Netherlands
Tel. (013) 673973, TELEX 52237
Phone or write for the name of your local Fluke representative.

©1982, John Fluke Mfg. Co., Inc., all rights reserved

Printed in U.S.A.

B0091A-10U8005/SE EN

FLUKE®

Technical Data

Application Information B0101

1720A RS-232-C Interfacing To Serial Printers

RS-232-C Defined

EIA Standard RS-232-C provides the electronics industry with the ground rules necessary for independent manufacturers to design and produce both data terminal and data communication equipment that conforms to a common interface requirement. As a result, a data communications system can be formed by connecting an RS-232-C data terminal to an RS-232-C data communication peripheral (such as a TTY, MODEM, computer, etc.).

The RS-232-C is a hardware standard which guarantees the following:

1. Each device on RS-232-C will use a standard 25-pin connector which will mate to another standard 25-pin of opposite sex.
2. No matter how the cables are connected, no smoke or damage will occur.
3. The data and handshake lines will each be given a specific name.

RS-232-C data and handshake lines:

In serial communications, control and data signals usually come from one pair of lines; an additional line sometimes provides a busy signal - used to delay data transmission until the device can handle that data. The data and handshake lines in RS-232-C send information uni-directionally (simplex); that is, one end of a cable transmits data or handshake and the other end receives data or handshake. Care must be taken to insure that each wire in RS-232-C has the appropriate transmitter and receiver combination. Transmitters connected to transmitters and receivers connected to receivers provide no data communication. To alleviate this problem, the RS-232-C Standard calls out the interface on one end of the cable to be designated as a "Terminal" and the interface on the other end is "Data Communication Equipment". The standard defines the data handshake signals on each pin of the connector for the "Data Communication Equipment" and the "Terminal".

Note: *There is a glossary at the end of this bulletin which can be referred to for most of the terms which may be unfamiliar to the reader.*

RS-232-C Signal Considerations

Timing format conforming to asynchronous operation is shown in figure 1.

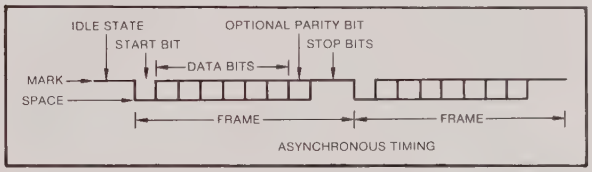


Figure 1. RS-232-C Timing Formats

Asynchronous Operation:

In asynchronous operation each character is bracketed by start and stop bits. These bits separate the characters and synchronize both the transmission and reception of data. When data is not being sent, the transmit line is held high (High=1).

Transmission Mode:

Transmission mode is a fundamental system requirement. It defines the communication ability of both instruments in the system configuration. SIMPLEX indicates data transmission in one direction only. HALF-DUPLEX permits two way communication, but not simultaneously. Simultaneous transmission of data in both directions defines the FULL-DUPLEX system. Obviously, an instrument capable of full-duplex operation can be down graded to simplex operation. However, the reverse is not possible without degrading the system capability.

Baud Rate:

Baud rate is usually selectable on the RS-232-C Interface. If it is not, the manufacturer usually offers a choice when the instrument is purchased. Character format (bits per character and parity) is somewhat flexible between instruments. Investigate the requirement of both instruments before committing either to a system configuration.

All data, documentation, dialog, diagrams, suggestions, reports and/or other forms of media contained in this bulletin are intended to be informational in nature only. Implementation of such data to a user's application should ONLY be made after careful analysis by the user's own software experts. **John Fluke Mfg. Co., Inc., specifically disclaims all warranties on such information, express or implied, including but not limited to any warranty of merchantability, fitness, or adequacy for any particular purpose or use.**

FLUKE®

Data Interface Levels:

The 1720A uses EIA voltages for data interface levels. EIA voltage levels are: 1 or OFF=−25 to −3V dc, 0 or ON=+3 to +25V dc.

This works fine on paper. However, in practice the user must be aware of the subtleties of serial binary data interchange to ensure that any two pieces of RS-232-C equipment will be compatible.

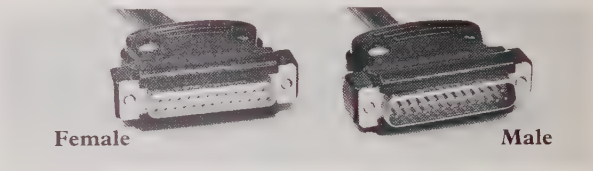
RS-232-C Specification

You can obtain information on this specification from the Electronic Industries Association, Engineering Department, 2001 1st N.W., Washington, D.C. 20006. Send $6.90 per document copy and ask for EIA Standard RS-232: "Interface Between Data Terminal Equipment and Data Communications Equipment Employing Serial Binary Data Interchange". A companion document, "Industrial Electronics Bulletin No 9 - Application Notes for EIA Standard RS-232-C", costs $3.50.

Printer Considerations for RS-232-C and the 1720A Instrument Controller

The 1720A Instrument Controller has two serial interface ports that meet the requirements of RS-232-C for full duplex asynchronous communication using the EIA voltage levels. The user must verify that the data communication peripheral is pin-for-pin compatible to the 1720A. Table 1 details the RS-232-C cable pinouts for the 1720A and some common printers.

The Y1709 Printer cable has been configured to work directly with the 1776A Printer.The cable has a male connector which mates with the female connector on the 1776A printer. The cable's female connector mates with the male connector on the 1720A. Once the cable has been connected to the 1776A/1720A, the system is ready to be powered up and begin printing. Print statements are given later in this bulletin.



RS-232-C Connectors

RS-232-C Compatibility Problems

Typical compatability problems associated with RS-232-C are:

1. There are no software standards associated with RS-232-C. Many types of communication protocols serve RS-232-C systems. One protocol uses USASCII code STX (start of text) to precede data and ETX (end of text) to follow data transmission. Another uses USASCII code STX (start of text) to precede data and ETX (end of text) to follow data transmission. Another uses USASCII ACK to acknowledge message receipt and NAK to indicate no acknowledgement. This ACK/NAK combination is usually found in polling computer configurations. (STX, ETX, ACK and NAK are nonprinting characters, for "handshaking" or control only).

Table 1. 1720A TO PRINTER CABLE CONNECTIONS

1720A RS-232-C Signal Definitions		Fluke 1776A Tally T1605 T1612	TI 800 Series and Centronics 704	DEC LA120	Lear Siegler 300 Series and Data Royal 5000 7000	IDS 440	Facit 4555 and Anadex DP8000 and Printronix 300	Fluke 2020A	Teletype 43	Printer Signal Definitions
Chassis Ground (Shield)	1	1	1	1	1	1	1	—	1	Chassis Ground (Shield)
Transmitted Data	2 →	3	3	3	3	3	3	2	3	Received Data
Received Data	3 ←	2	2	2	—	—	—	—	—	Transmitted Data
Request to Send	4 →	—	8	—	—	—	—	—	—	Line Signal Detector
Clear to Send	5 ←	—	—	—	—	—	—	—	—	
Data Set Ready	6 ←	20	20	20	20	—	—	—	—	Data Terminal Ready
Signal Common	7 ↔	7	7	7	7	7	7	7	7	Signal Common
Secondary Channel Receive	12 ←	11 or 19	11	11 or 19	19	20	20	6	—	Printer Busy/Ready
Data Terminal Ready	20 →	—	6	—	—	—	—	—	—	Data Set Ready
Cable Connector Type		Male	Male	Female	Male	Female	Male	Male	Female	Female = DB25S Male = DB25P
Comments			TI820 requires a parameter 14- to handle busy							
Fluke Cables		Y1709	Y1709	Y1705 & Y1707	Y1709	N/A	N/A	N/A	Y1705 & Y1707	

©1980, John Fluke Mfg. Co., Inc., all rights reserved

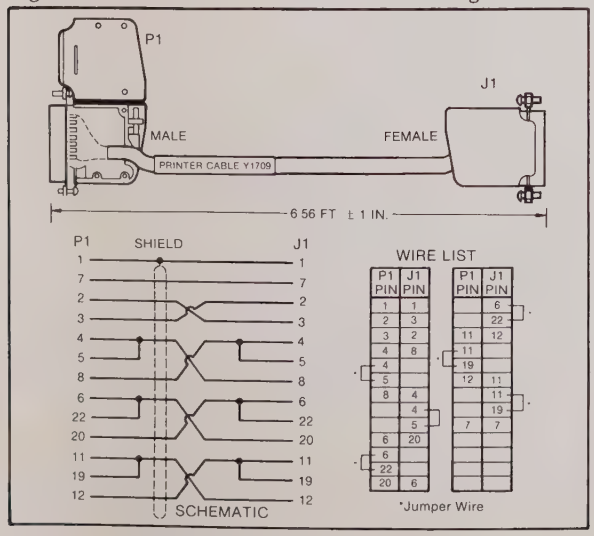
2. RS-232-C terminals and RS-232-C data communications equipment are not always hardware compatible.

   A. For example, the two instruments must share at least one of the features from each of the following characteristics:

      1.) Timing Format - Synchronous or asynchronous
      2.) Transmission Mode - Simplex, half-duplex, or full duplex
      3.) Baud Rate (bits per second) - 75, 110, 134.5, 150, 300, 600, 1200, 1800, 2000, 2400, 3600, 4800, 7200, 96000, 19200
      4.) Bits per character - 5, 6, 7, or 8
      5.) Parity Bit - Odd, even, high, low, not used
      6.) Data Interface Levels - EIA voltage levels or 20 mA current loop

   B. Care must be taken to ensure that the RS-232-C cable is correct for the application. One of the ambiguous areas in an RS-232-C connection is the use of pin 2 for transmitted data (TD) and pin 3 for received data (RD). The confusion arises in a simplex or half-duplex connection, where pin 2 at one end of the line must go to pin 3 at the other end, and vice versa; this pin transposition can be handled in the cable itself or at either connector. Another confusing aspect of the RS-232-C standard is the pin used to indicate a busy condition. Occasionally pin 11 - normally unassigned - has this task; in other cases pin 19 or pin 20 - with the appropriate polarity - is used.

Figure 2. RS-232-C/1720A Printer Cable Diagram



Consult the manual for a particular device to determine the proper cabling. However, if the manual is not available, the following simple test will tell you if a device is a terminal or data communication device (MODEM) in most cases.

Compatability Test:
RS-232-C Cable Application

Measure voltage at pins 2 and 3 with ground lead connected to pin 7.

PERFORM TEST WITH NO CABLES CONNECTED
"TERMINAL"
Pin 2 < −3V Pin 3 0 to +2V Pin 7 GROUND
"DATA COMMUNICATIONS DEVICE" (MODEM)
Pin 2 0 to +2V Pin 3 < −3V Pin 7 GROUND

Obtaining A Printed Program Listing From The 1720A

RS-232-C Printer:
Select an RS-232-C port, connect the printer and set appropriate baud rates

Basic Immediate Mode :

1. Type: OLD "filename" <CR>
2. Type: SAVE "KB1:" <CR>
   or SAVE "KB2:" <CR>

File Utility Program:

Type KB1:=file name <CR>
KB2:=file name <CR>

Note: *Only ASCII files (.BAS) can be listed in FUP.*

Example:

1720A BASIC program which inputs data from the floppy disk and outputs this data to an RS-232 printer.

```
10 ' PROGRAM      :  LIST.BAS
20 ' PROGRAMMER   :
30 ' REVISION     :  1.0
40 ' DATE         :  15 APR 1980
50 '
60 ' DESCRIPTION :  LISTS AN ASCII FILE ON A RS-232-C PRINTER
70 '
80 '
90                          ON ERROR GOTO 280
100 '
110 '                                          ! ESCAPE SEQUENCE
120 ES$ = CHR$(27%) + "E"                      ! RESET SCREEN
130 PRINT ES$;'1p';                            ! PROMPT
140 PRINT ES$;'4;ENTER FILENAME TO BE LISTED' ! STORE FILENAME
150 INPUT A$
160 '                                          ! OPEN CHANNEL FOR INPUT
170 CLOSE 1 \ OPEN A$ AS FILE 1                ! OPEN KB1: FOR OUTPUT
180 CLOSE 2 \ OPEN 'KB1:' AS NEW FILE 2
190 '                                          ! TOP OF FORM
200 PRINT #2,CHR$(12%);
210 '                                          ! INPUT EACH LINE
220 INPUT LINE #1, A$                          ! PRINT EACH LINE
230 PRINT #2, A$                               ! GET NEXT LINE
240 GOTO 220
250 '
260 '                ERROR HANDLER
270 '                                          ! FILE DOESN'T EXIST
280 IF ERR = 305 THEN RESUME 310               ! END OF FILE
290 IF ERR = 307 THEN RESUME 330
300 PRINT 'ERROR';ERR;'@ LINE';ERL; \ RESUME 330
310 PRINT ES$;'6;HFILE DOESN'T EXIST - TRY AGAIN'; \ GOTO 140
320 '
330 CLOSE 1,2
340 END
```

Glossary

Asynchronous Transmission	Having a variable time interval between successive bits, characters, or events
Baud Rate	The number of bits that can be transmitted in one second
Bit	One of the characters of a two valued or binary number system such as 0 and 1. A bit has come to signify the smallest piece or smallest unit of information. (A single pulse in a group of pulses.)
EIA	Electronic Industries Association 2001 Eye Street, Northwest Washington, D.C. 20006
Format	The predetermined arrangement of characters
Frame	A time period encompassing the bits which define a character
Full Duplex	Simultaneous communication between two points in both directions
Half Duplex	One way communication between two points in either direction
Handshaking	Communication which takes place between two devices for the purpose of informing each other about the status of data being transmitted, received or processed, in order that this may be done in a cooperative, orderly and timely manner without errors. Handshaking is vital to the operation of asynchronous transmission.
Interface, Electrical	Electrical interconnection between system elements
I/O	Input/Output
Modem	A contraction of "modulator-demodulator". In the modem, the square-edged pulse train is impressed (modulated) on a carrier signal of a frequency which is within the telephone channel frequencies between 300 and 3300 Hz. The modem also extracts (de-modulates) the square-edged pulse train from the carrier wave allowing bit-serial communication over standard telephone lines.
Parity	Proper value in the binary check digit of the transmitted and received data.
Parity Bit	A binary digit appended to an array of bits to make the sum of all bits always odd or always even.
Parity Checking	A method used to detect single bit errors
Peripheral Equipment	In a data processing system, any equipment distinct from the central processing unit that may provide the system with outside communication or additional facilities
RS	Recommended standard
Simplex	Data transmission in one direction only
Sync	Short for synchronization
Sync Pulse	An electrical pulse transmitted to a circuit by the master equipment to operate a slave in synchronism with the master.
Synchronous Transmission	A precisely timed bit stream and character stream
TTY	Teletypewriter Equipment

FLUKE®

John Fluke Mfg. Co., Inc.
P.O. Box 43210, Mountlake Terrace, WA 98043
800-426-0361 (toll free) in most of U.S.A.
206-774-2481 from AK, HI, WA and Canada
206-774-2398 from other countries

Fluke (Holland) B.V.
P.O. Box 5053, 5004 EB, Tilburg, The Netherlands
Tel. (013) 673973, TELEX 52237
Phone or write for the name of your local Fluke representative.

Printed in U.S.A.
B0101B-10U8203/SE EN

DISK HOLDER
P/N 632984

www.manualsplus.com
1029149